Optimal Viewport-Adaptive 360-degree Video Streaming against Random Head Movement

Han Hu, Zhimin Xu, Xinggong Zhang, Zongming Guo*
Institute of Computer Science & Technology, Peking University
Beijing, P.R. China
{huhan951753, zhiminxu, zhangxg, guozongming}@pku.edu.cn

***Abstract*—Recently, a significant interest in 360-degree virtual reality (VR) video has been formed. However, a key problem is how to design a robust adaptive streaming approach and implement a practical system. The traditional streaming methods which are not sensitive to user's viewport could cause huge bandwidth budget with low video quality, while the viewport-adaptive schemes may be not accurate enough especially under random head movement. In this paper, we have designed an optimal viewport-adaptive 360-degree video streaming scheme, which is to maximize Quality of Experience (QoE) by predicting user's viewport with a probabilistic model, prefetching video segments into the buffer and replacing some unbefitting segments. In this way, continuous and smooth playback, high bandwidth utilization, low viewport prediction error and high peak signal-to-noise ratio in the viewport (V-PSNR) can be obtained. In order to deal with user's head movement, we have reduced prediction error rate by employing a probabilistic viewport prediction model, and a replacement strategy has been applied to update the downloaded segments when the user's viewport suddenly changes. To reduce smoothness loss, the segments' oscillation during playback has been considered. We also developed a prototype system with our method. The well-designed experiments provided numerous results which proved the better performance of our scheme.**

***Index Terms*—Quality of Experience (QoE), 360-degree Video Streaming, Virtual Reality (VR), Dynamic Adaptive Streaming over HTTP (DASH), Probabilistic Viewport Prediction**

I. Introduction

Nowadays, the 360-degree technology has been reforming the video industry. 360-degree videos are very prevalent in large number of users on major video platforms such as Facebook and YouTube [1]. The demand on better user experience while watching 360-degree videos is sharply increasing, and virtual reality (VR) techniques have become more and more significant owing to this trend [2].

However, it is hard to deliver 360-degree videos with the highest quality to the end users by the current popular technologies, since the 360-degree videos usually have an extremely high resolution and cannot be transferred smoothly under normal network conditions [3]. 360-degree videos are often limited by the high bandwidth requirement, which results in low quality [4]. Several approaches have been proposed to solve this problem, including using the server push feature of the HTTP/2 standard to deliver the video, and the viewport-dependent solutions are considered to be good candidates [5].

It is spotted that only a portion of the video is viewed at a time, so only transmitting partial contents of user's Field of View (FOV) seems to be a nice idea [6]. It becomes a critical issue to predict user's viewport for 360-degree video streaming [7], and wrong prediction caused by the randomness of user's head movement will heavily reduce the Quality of Experience (QoE). Some researchers have attempted to predict the FOV [8], but the performance of their model is not good enough, especially for long-term prediction (over 5 seconds), which leads to relatively low accuracy.

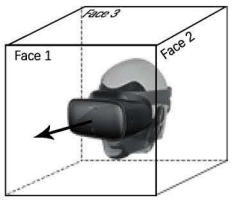
As for encoding and transforming, there are mainly two categories of viewport adaptive streaming: tile-based and asymmetric panorama. These works [3]–[8] apply tile-based methods, which crop 360-degree videos into multiple tiles (or blocks) in space, then partition and encode the tiles into multi-bitrate segments. Although the tile-based methods could provide high flexibility to request an interesting area, they may arise black area if the user's head movement has been incorrectly predicted. What's more, the tile-based methods need support of High Efficiency Video Coding (HEVC) and is hard to implement over most devices and frameworks [9].

On the contrast, the asymmetrical adaptive streaming methods, such as Qualcomm's Truncated Pyramid Projection (TSP) [10] and Facebook's offset cubemap [11], can solve the problems above to some extent. Different from the traditional projection schemes like the Equirectangular projection (ERP) [12], these methods transform 360-degree videos into viewport-dependent multi-resolution panorama. By sacrificing some bandwidth utilization, they can maintain the quality of the viewport at a high level [13]. However, these methods still face the problem of user's random head movement. If wrong segments with incorrect viewport had been prefetched, users would view an area with low resolution as well as low quality. This drops QoE significantly [14].

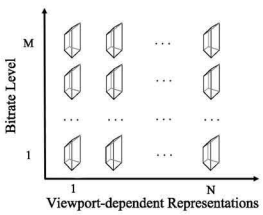
In this paper, we have designed an optimal viewport-adaptive 360-degree video streaming scheme, adopting probabilistic viewport adaptation and buffer replacement strategy to deal with prediction error caused by the randomness of user's head movement. Firstly, a probabilistic viewport prediction model has been employed to predict user's future orientation with a probability distribution, as well as increase

*Corresponding author.
This work was supported by National R&D project of China under contract No. 2018YFB0803702, Ali AIR Funding No. XT622018001708.

978-1-5386-8088-9/19/$31.00 ©2019 IEEE



(a) Faces of a Cube Being Viewed



(b) Different Segments

Fig. 1. Projection and Transformation

the accuracy of viewport prediction. Then we build a buffer-based model to store the prefetched segments and calculate the expected quality and smoothness loss, both of which have a great impact on QoE of adaptive streaming [15]. Since the network conditions are always fluctuating, and sometimes the user's orientation becomes too random to exactly predict, the prefetched segments may be out of date with incorrect viewport. To keep providing high quality 360-degree video to the user, we utilize a replacement strategy so as to update the downloaded segments which are no longer the best choices when the network conditions and user's viewport make sudden change. At last, we have designed an optimization problem at every download step that leads the client to decide which segments to prefetch and which to replace.

We have carried out extensive experiments under controlled test-bed and different network conditions with a dataset of head movement traces [7] to evaluate our scheme. We compare our scheme (PR-TSP) with some other methods, including transmission in the form of ERP without viewport adaptation (ERP), transmission in the form of TSP without probabilistic viewport prediction (TSP), and TSP with probabilistic viewport prediction but without the segment replacement strategy (P-TSP). Our approach outperforms the other methods under real Internet bandwidth trace, achieving about 26% higher peak signal-to-noise ratio in the viewport (V-PSNR) than ERP, 6% higher than TSP and 3% higher than P-TSP. It also achieves fewer stall times and lower prediction error under various network conditions and head movement traces.

The rest of paper is organized as follows. Section II specifically presents our probabilistic viewport prediction model, buffer-based model, replacement strategy and optimization problem. In Section III, we illustrate the architecture of our 360-degree video adaptive streaming system, reveal our experiments and analysis the results to evaluate the different methods' performance. And eventually we come to the conclusion in Section IV.

II. SCHEME DESIGN

In this section, we introduce our optimal viewport-adaptive 360-degree video streaming scheme. This scheme utilizes a relatively reliable probabilistic model of viewport prediction. It also considers the buffered segments as a whole in order to maximize the expected QoE.

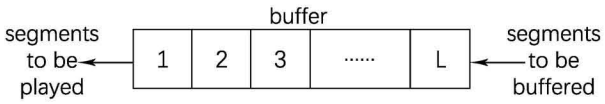


Fig. 2. The Organization of Buffered Segments

A. Problem Formulation

For every 360-degree virtual reality (VR) video, we project it into a cube and define K faces depending on the user's viewport, as shown in Fig. 1(a). Then we apply TSP and transform it into $N \times M$ different segments, which represent N kinds of viewport-dependent representations and M kinds of bitrate levels of each representation, as shown in Fig. 1(b).

The buffer is organized in this way. We assume that the buffer can store up to L different segments as shown in Fig. 2, which are numbered from 1 to L as the sequence of being played. While the user is watching a video segment, segments in the buffer would be played in numbered order. Then the buffer is divided into L parts, which are different in time sequence. Each position can choose one of $N \times M$ different segments described in Fig. 1(b) to download and buffer. We make prefetching plan and download the most suitable segments according to bandwidth and user's viewport, so as to get the highest QoE of the whole buffer. At the following download steps, the segments which are not the best choices based on the real-time bandwidth and viewport will be abandoned with new versions selected instead.

At each download step, we use matrices to describe our prefetching plan of video segments, as well as the state of buffer. Let $i \in \{1...N\}$ denote the different viewport-dependent representations, $j \in \{1...M\}$ denote the different bitrate levels, and $l \in \{1...L\}$ denote the sequence number of the locations in the buffer. Then we define the set of streaming segments as matrix $\mathbf{X} = \{x_{ijl}\}$, while $x_{ijl} = 1$ means that for the l-th location of the buffer, we choose to download the segment of the i-th viewport at j-th bitrate level, or segment(i,j,l), $x_{ijl} = 0$ otherwise. Our final target is to decide $\mathbf{X}$ to maximize QoE. In order to quantify it, we define matrix $\mathbf{Y} = \{y_{ijl}\}$ to describe the state of the buffer, while $y_{ijl} = 1$ means that segment(i,j,l) has been bufferd and $y_{ijl} = 0$ otherwise. By deciding $\mathbf{X}$, the buffer gets initialized and updated, and the unbefitting video segments can be replaced in time.

To predict user's viewport, we also define the viewing probability of each face. For the l-th location in the buffer, we define P_l^k as the normalized viewing probability of the k-th face described in Fig. 1(a) and D_{ijl}^k as the average distortion on the k-th face of segment(i,j,l). The method of calculating P_l^k will be mentioned in Section II-C3. We also define R_l as the total bitrate budget at the time of the l-th position and r_{ijl} as the bitrate of segment(i,j,l).

To maximize the quality of viewport adaptive streaming, we define two QoE functions: 1) expected distortion of the whole buffer $Sum(\mathbf{X})$, which quantifies the increment of the quality distortion of the buffer under the consideration of viewing probability of all the segments to be prefetched. 2)

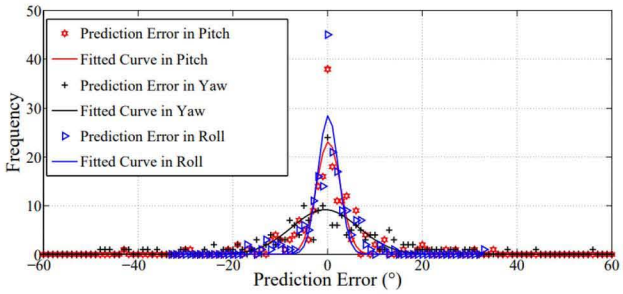


Fig. 3. Prediction Error Gaussian Distribution when δ = 3s

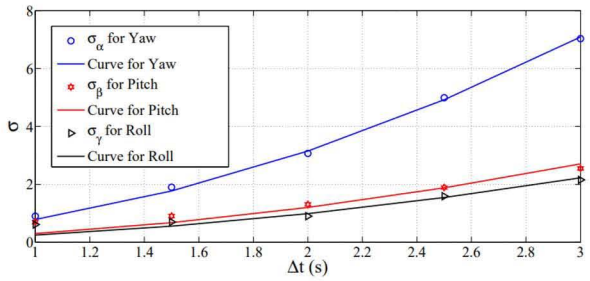


Fig. 4. Variation of σ_α, σ_β and σ_γ

spatial quality variance $Dev(\mathbf{X})$, which denotes the quality smoothness loss of the segments. At each download step, our goal is to minimize the weighted distortion of these two functions. With the definition of ρ as the weight of $Sum(\mathbf{X})$, our optimization problem can be formulated as:

$$\min_{\mathbf{x}} \rho \cdot Sum(\mathbf{X}) + (1-\rho) \cdot Dev(\mathbf{X})$$

$$s.t. \sum_{j=1}^{M} x_{ijl} \leq 1, x_{ijl} \in \{0,1\}, \forall i, \forall l,$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M} u(x_{ijl} - y_{ijl}) r_{ijl} \leq R_l,$$

where

$$u(z) = \begin{cases} 1, z > 0 \\ 0, z \leq 0 \end{cases}$$

The first constraint in the optimization problem gives the restriction on x_{ijl}. For each part, it only needs to select at most one bitrate level of each viewport representation.

The second constraint restricts the total bitrate of selected segments. The total bitrate budget R_l for each part can be estimated by any rate adaptation algorithm of Dynamic Adaptive Streaming over HTTP (DASH). We use the function $u(z)$ to show that only when segment(i,j,l) has been chosen ($x_{ijl} = 1$) and has not been buffered ($y_{ijl} = 0$) should it be downloaded, which occupies part of R_l.

B. Expected Distortion and Spatial Quality Variance

To define the expected distortion of the whole buffer, firstly we calculate the expected distortion at the l-th location as:

$$E(D_l) = \sum_{i=1}^{N}\sum_{j=1}^{M}(x_{ijl} - y_{ijl}) \sum_{k=1}^{K} D_{ijl}^k P_l^k \quad (1)$$

Then the expected distortion of the whole buffer would be quantified as:

$$Sum(\mathbf{X}) = \sum_{l=1}^{L} \lambda_l E(D_l) \quad (2)$$

where λ_l is a parameter that decreases with l. We assume that the duration of each segment is T, then a larger l means that the segment is later to be played, so it would be given less weight λ_l. Having found out the relation between the standard deviation of prediction error and prediction duration in Section II-C2, we decide that λ_l should be set as:

$$\lambda_l = \frac{1}{\sqrt{l \cdot T}} \quad (3)$$

To take the quality smoothness into account, we calculate the average distortion of the whole buffer:

$$E(\overline{D}) = \frac{1}{L}\sum_{l=1}^{L} E(D_l) \quad (4)$$

Then we determine the standard deviation of distortion, which is suitable to denote the quality smoothness loss:

$$Dev(\mathbf{X}) = \sqrt{\frac{\sum_{l=1}^{L}(E(D_l) - E(\overline{D}))^2}{L}} \quad (5)$$

C. Probabilistic Model of Viewport Prediction

In order to prefetch the video segments which is most likely to be viewed, we should predict the user's viewport and deal with prediction errors. In this section, we build a probabilistic model to solve this problem.

1) Linear Regression Prediction of Orientation: We denote the user's orientation (Euler angle) as yaw (α), pitch (β) and roll (γ). The slopes of the trend are denoted as v_α, v_β and v_γ, which are calculated by Least Squares Method (LSM) according to the historical samples. We define t_0 as the current time of system, then the Euler angle after δ can be predicted with Linear Regression (LR) strategy as:

$$\begin{cases} \hat{\alpha}(t_0+\delta) = v_\alpha \delta + \alpha(t_0) \\ \hat{\beta}(t_0+\delta) = v_\beta \delta + \beta(t_0) \\ \hat{\gamma}(t_0+\delta) = v_\gamma \delta + \gamma(t_0) \end{cases} \quad (6)$$

In particular, owing to the assumption that the duration of each segment is T, for the l-th location in the buffer defined in Section II-A, the relation of δ and l is $\delta = l \cdot T$.

2) Distribution of Prediction Error: We use one real head movement trace, and plot the prediction error of the LR method in Fig. 3. It suggests that the prediction error follows Gaussian Distribution, and the parameter μ and σ can be learned by curve fitting. Thus the probability distributions of α, β and γ can be derived as:

$$\begin{cases} P_{yaw}(\alpha) = \frac{1}{\sigma_\alpha\sqrt{2\pi}} exp(-\frac{(\alpha-(\hat{\alpha}+\mu_\alpha))^2}{2\sigma_\alpha^2}) \\ P_{pitch}(\beta) = \frac{1}{\sigma_\beta\sqrt{2\pi}} exp(-\frac{(\beta-(\hat{\beta}+\mu_\beta))^2}{2\sigma_\beta^2}) \\ P_{roll}(\gamma) = \frac{1}{\sigma_\gamma\sqrt{2\pi}} exp(-\frac{(\gamma-(\hat{\gamma}+\mu_\gamma))^2}{2\sigma_\gamma^2}) \end{cases} \quad (7)$$

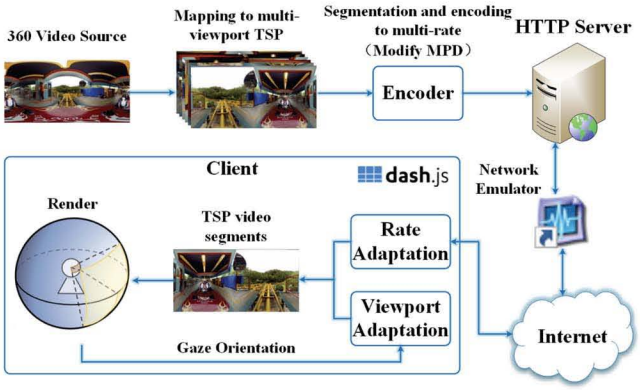


Fig. 5. System Architecture

Moreover, it is reasonable to assume that long-term prediction would result in larger error deviation. Therefore, we also plot the standard deviations against prediction duration δ in Fig. 4. It can be spotted that the standard deviation of prediction error σ is strictly increasing with the square of prediction duration δ^2, so we take this effect into consideration while calculating expected distortion in Section II-B.

Since yaw, pitch and roll are independent of each other, the correct probability that an Euler angle (α, β, γ) exactly fits the user's orientation is calculated as:

$$P_E(\alpha, \beta, \gamma) = P_{yaw}(\alpha) P_{pitch}(\beta) P_{roll}(\gamma) \tag{8}$$

3) Viewing Probability of One Face: According to the probability of user's orientation, we can obtain the viewing probability of each face shown in Fig. 1(a). A spherical point is defined as (φ, θ). Since these spherical points could be distributed among K different TSP's faces, we define $L_k(\varphi, \theta)$ as the set of points within the k-th face. To simplify the problem, we let the viewing probability of k-th face P^k equal to the average probability of orientations in $L_k(\varphi, \theta)$ as:

$$P^k = \frac{1}{|L_k(\varphi, \theta)|} \sum_{(\alpha,\beta,\gamma) \in L_k(\varphi,\theta)} P_E(\alpha, \beta, \gamma) \tag{9}$$

III. System Design and Evaluation

To evaluate the performance of our strategy, we have implemented a practical prototype system.

Unlike the tile-based methods [3]–[8], our proposed scheme can be easily applied into a 360-degree video adaptive streaming system. Fig. 5 shows the system architecture we have designed. The most important modules are the Rate Adaptation module and the Viewport Adaptation module, which adopt our adaptation algorithms described in Section II to help the client choose the best video segments stored at the HTTP server.

We also carry out extensive simulation experiments and real Internet experiments under various head movement traces.

A. Setup

In the experiments, we imitate user's head motion by embedding real user's head movement trace into the player. We also set the network conditions to observe how different schemes react to the network fluctuations. Specially, we examine the performance on video sequence and 5 user's head movement traces on this video, which are generously provided by AT&T [7]. The sequence is about 3-4 minutes long with the resolution 2880×1440 in ERP format. For each viewport-dependent representation, it is further partitioned into segments with the same duration 1 second. The bitrate levels of each segment are set as {300kbps, 700kbps, 1500kbps, 2500kbps, 3500kbps}. The video codec is the widely used open source encoder x264 [16]. All the video segments are packaged by the tool MP4Box [17].

TABLE I
Performance on Fixed Bandwidth

Methods	ERP	TSP	P-TSP	PR-TSP
Average Bandwidth (kbps)	1500.0	1500.0	1500.0	1500.0
Stall (Times)	0	0	0	0
Viewport Error (Times)	0	12	8	3
Average V-PSNR (dB)	33.18	39.30	39.87	40.10

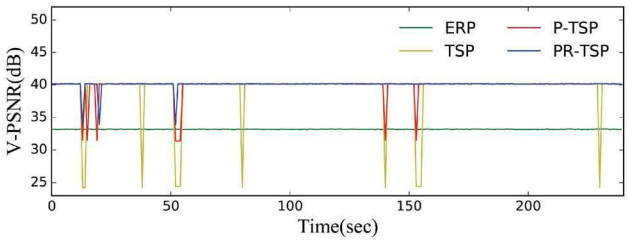


Fig. 6. Bitrates and V-PSNR under Fixed Bandwidth

B. Experiment Results and Analysis

To validate the efficiency of the proposed scheme, we select three typical 360 video streaming methods as the comparisons:

- ERP: This method treats 360-degree video streaming as ordinary video. It is widely developed on major video platforms like YouTube [1].
- TSP: This method uses LR method to predict future viewport and request corresponding segments, such as in [18]. But it neither applies probabilistic viewport prediction nor utilizes a buffer to consider the overall quality. This is the baseline of probabilistic viewport adaptive streaming. We use it for comparison to evaluate the performance of the probabilistic viewport prediction.
- P-TSP: This method has all the features of our proposed scheme, including probabilistic viewport prediction, buffer-based model and optimal segments prefetch. The only difference is that this method doesn't replace the downloaded segments in the buffer, even if a different prefetching plan has been made according to the current network conditions and viewport at a new download step. By comparing with this method, we can evaluate the necessity of our replacement strategy.
- PR-TSP: This is our proposed method. It utilizes probabilistic viewport prediction, buffer-based model and optimal segments prefetch. The buffer size (L) is fixed to a suitable value. It also uses a replacement strategy when the viewport and network conditions have changed, so as to choose better segments instead of the buffered ones.

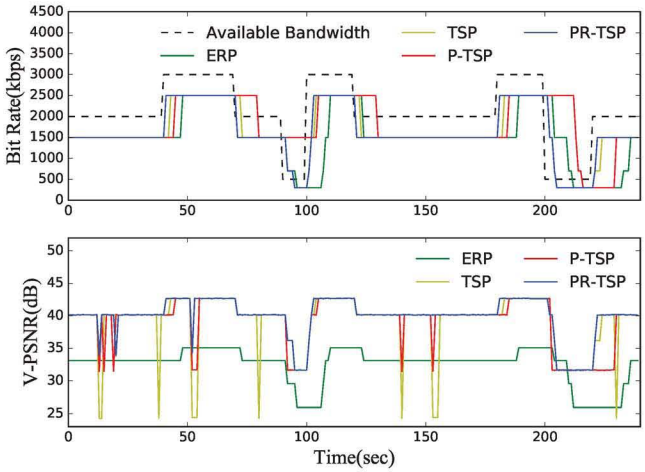


Fig. 7. Bitrates and V-PSNR under Long-term Bandwidth Variations

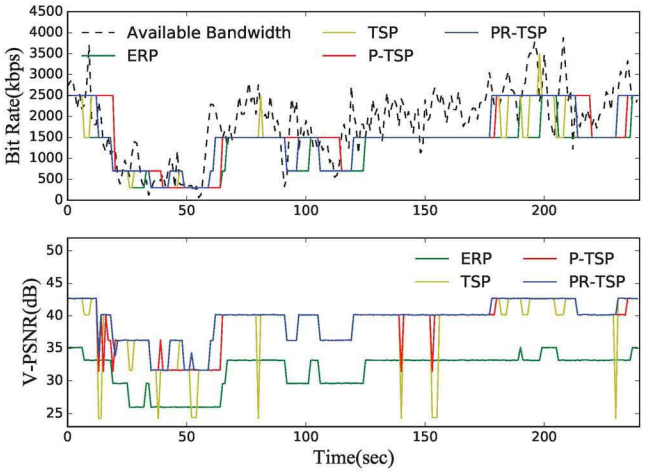


Fig. 8. Bitrates and V-PSNR under a Real Internet Bandwidth Trace

TABLE II
PERFORMANCE ON LONG-TERM BANDWIDTH VARIATIONS

Methods	ERP	TSP	P-TSP	PR-TSP
Average Bandwidth (kbps)	1531.7	1646.7	1790.0	1657.5
Stall (Times)	8	5	13	4
Viewport Error (Times)	0	12	8	3
Average V-PSNR (dB)	32.47	38.87	39.18	40.10

TABLE III
PERFORMANCE ON REAL INTERNET TRACE

Methods	ERP	TSP	P-TSP	PR-TSP
Average Bandwidth (kbps)	1256.7	1447.5	1562.5	1503.3
Stall (Times)	2	7	19	1
Viewport Error (Times)	0	12	8	3
Average V-PSNR (dB)	31.63	38.45	38.80	39.73

In performance comparison, we take the following measurement metrics into consideration:

- Stall: This metric evaluates the playback continuity by counting the times of rebuffering events over the total video streaming time.
- V-PSNR: This metric directly indicates the quality of content in the user's viewport.
- Viewport Error: This metric records the incorrect times of prefetching segments due to viewport prediction error.

1) Experiments with Fixed Bandwidth: Firstly, we implement experiments under fixed bandwidth (2000kbps) to check out the performance of the mentioned methods in stable network conditions. We use just one head movement trace to control variables, and replay it for all of them.

Fig. 6 shows the V-PSNR of the requested video segments when the methods made their prefetch plan per second. As they use the same bandwidth estimation method, all the methods always choose to prefetch the segment with a bitrate of 1500kbps, which is the closest to the bitrate budget, just as what Table I records. However, the differences appear when V-PSNR is calculated. ERP has the lowest viewport quality since it transfers the whole 360-degree picture. As for the other schemes, they perform almost the same when they accurately predict the user's head movement and viewport, but the viewport quality sharply decreases when the prediction is not precise enough. Without a probabilistic model, TSP tends to make mistakes most frequently and cannot maintain a high accuracy. Once a segment has been downloaded, P-TSP does not replace it even if the user's viewport has changed, which leads to some extra mistakes. With the probabilistic model and buffer replacement strategy, our PR-TSP achieves a better prediction accuracy and gets higher V-PSNR.

We calculate and record the average bandwidth, stall times, viewport error and average V-PSNR in Table I. The ERP method does not predict the user's viewport, so it never make mistakes, but it acquires the lowest average V-PSNR. The proposed PR-TSP outperforms others on these two subjects.

2) Experiments with Long-term Bandwidth Variations: In order to evaluate the performance of the methods in relatively unstable network conditions, we carry out experiments under long-term bandwidth variations. The bandwidth varies from 500kbps to 3000kbps, and it lasts for a period of time at a certain level. We choose the same head movement trace in Section III-B1.

The results are shown in Fig. 7. PR-TSP performs better in prefetching segments since it utilizes the segment replacement strategy, though all the four schemes adopt the same rate-adaptation method. It also obtains higher V-PSNR compared with other methods. P-TSP tends to farthest deviate from the real bandwidth trace and is not sensitive to bandwidth changes because it never updates the segments that are decided with the prediction of earlier network conditions, which leads to rebuffering events (at around $t = 80s$, etc) and waste of bandwidth (at around $t = 225s$, etc).

As listed in Table II, P-TSP acquires the highest average bandwidth but its stall is terribly frequent. Relatively, the proposed PR-TSP works better, which also outperforms the other methods according to the number of viewport error and average V-PSNR.

3) Experiments with Real Internet Trace: We also implement experiments with a real Internet trace so as to evaluate the performance of the four methods in realistic network conditions. The trace is selected from [19]. The head movement trace is still the same.

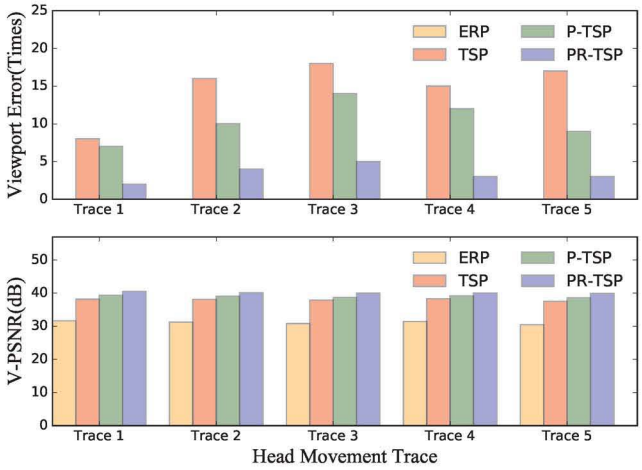


Fig. 9. Average V-PSNR and Viewport Error on Five Different Head Movement Trace

The results displayed in Fig. 8 suggests that the proposed PR-TSP fits the best with the available bandwidth while prefetching video segments, as ERP tends to select segments with lower bitrate and TSP may face greater smoothness loss. P-TSP gets obvious deviation for the same reason mentioned in Section III-B2. PR-TSP still performs well according to V-PSNR. Even if when all the methods except ERP make mistakes in predicting the user's viewport (at around $t = 13s$, etc), PR-TSP tends to get the smallest deviation as well as the least quality loss.

Table III shows a similar result with Table II. P-TSP increases its stall times since the network conditions become more complicated, but PR-TSP decreases it because the bandwidth doesn't drop so sharply as in Fig. 7. It reveals that our proposed PR-TSP outperforms other methods.

4) Experiments with Different Head Movement Traces: To more precisely evaluate the efficiency of viewport prediction strategy, we carry out experiments with five different head movement traces. The same real Internet trace as Section III-B3 is used in these experiments.

As shown in Fig. 9, our PR-TSP performs the best in the selected methods while considering viewport error and V-PSNR. For the former, it is unnecessary to concentrate on ERP since it delivers the whole 360-degree picture, and obviously PR-TSP achieves the smallest number of viewport error in the rest three schemes. For the latter, ERP gets the lowest average V-PSNR, while PR-TSP obtains the highest average V-PSNR, about 26% higher than ERP, 6% higher than TSP and 3% higher than P-TSP.

IV. CONCLUSION

In this work, we have designed an optimal viewport-adaptive 360-degree video streaming scheme to maximize QoE. A probabilistic model has been adopted to improve the accuracy of viewport prediction. To deal with prediction error caused by the randomness of user's head movement, we have applied a buffer-based model with a replacement strategy. Our optimization problem has taken the expected quality and smoothness loss into account so as to make a better prefetch plan. With the results of well-designed simulation experiments and real-world Internet experiments, we find out that this scheme can certainly obtain continuous and smooth playback, high bandwidth utilization, low viewport prediction error and high V-PSNR. In our future work, we plan to supplement this model to achieve better performance.

REFERENCES

[1] "Youtube live in 360 degrees encoder settings," https://support.google.com/youtube/answer/6396222, 2017.

[2] Xavier Corbillon, Francesca De Simone, Gwendal Simon, "360-Degree Video Head Movement Dataset," ACM, 2017, pp. 199-204.

[3] Wen Chih Lo, Ching-Ling Fan, Jean Lee, Chun-Ying Huang, Kuan-Ta Chen, Cheng-Hsin Hsu, "360 Video Viewing Dataset in Head-Mounted Virtual Reality," *ACM Multimedia Systems Conference*, ACM, 2017, pp. 211-216.

[4] Stefano Petrangeli, Viswanathan Swaminathan, Mohammad Hosseini, Filip De Turck, "Improving Virtual Reality Streaming using HTTP/2," *ACM Multimedia Systems Conference*, ACM, 2017.

[5] Stefano Petrangeli, Viswanathan Swaminathan, Mohammad Hosseini, Filip De Turck, "An HTTP/2-Based Adaptive Streaming Framework for 360 Virtual Reality Videos," *ACM Multimedia*, ACM, 2017.

[6] Ching-Ling Fan, Jean Lee, Wen Chih Lo, Chun-Ying Huang, Kuan-Ta Chen, Cheng-Hsin Hsu, "Fixation Prediction for 360 Video Streaming in Head-Mounted Virtual Reality," *The Workshop on Network & Operating Systems Support for Digital Audio & Video*, ACM, 2017, pp. 67-72.

[7] Feng Qian, Lusheng Ji, Bo Han, and Vijay Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges.* ACM, 2016, pp. 1-6.

[8] A. Mavlankar and B. Girod, "Video Streaming with Interactive Pan/Tilt/Zoom," *Signals and Communication Technology* 2010, pp. 431-455.

[9] "Tile Based HEVC Video for Virtual Reality," https://www.hhi.fraunhofer.de/en/departments/vca/research-groups/multimedia-communications/research-topics/tile-based-hevc-video-for-virtual-reality.html, 2018.

[10] E. Kuzyakov, "End-to-end optimizations for dynamic streaming," available online: https://code.facebook.com/posts/637561796428084.

[11] Patrice Rondao Alface, Jean Franois Macq and Nico Verzijp, "Interactive Omnidirectional Video Delivery: A Bandwidth-Effective Approach," *Bell Labs Technical Journal 16*, 2012, pp. 135-147.

[12] Sreedhar, Kashyap Kammachi, et al. "Viewport-adaptive encoding and streaming of 360-degree video for virtual reality applications," *Multimedia (ISM), 2016 IEEE International Symposium on*, 2016, pp. 583-586.

[13] Van der Auwera, Geert, et al. "AHG8: Truncated Square Pyramid Projection (TSP) For 360 Video," *Joint Video Exploration Team of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, JVET-D0071, 4th Meeting.* 2016.

[14] R. K. P. Mok, E. W. W. Chan and R. K. C. Chang, "Measuring the quality of experience of HTTP video streaming," *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops,* 2011, pp. 485-492.

[15] Liu Yitong, Shen Yun, Mao Yinian, Liu Jing, Lin Qi and Yang Dacheng, "A study on Quality of Experience for adaptive streaming service," *IEEE International Conference on Communications Workshops (ICC),* 2013, pp. 682-686.

[16] "x264,"available online: http://www.videolan.org/developers/x264.html, 2017.

[17] GPAC, "Mp4box,"available online: https://gpac.wp.imt.fr/mp4box.

[18] Masayuki Inoue, Hideaki Kimata, Katsuhiko Fukazawa, and Norihiko Matsuura, "Interactive panoramic video streaming system over restricted bandwidth network," *Proceedings of the 18th ACM international conference on Multimedia.* 2015, pp. 31-36.

[19] P. Vigmostad C. Griwodz H. Riiser, T. Endestad and P. Halvorsen, "Video streaming using a location-based bandwidth-lookup service for bitrate planning," *ACM Trans. Multimedia Comput. Commun. Appl (TOMC-CAP),* 2012, p. 24.