CUB360: EXPLOITING CROSS-USERS BEHAVIORS FOR VIEWPORT PREDICTION IN 360 VIDEO ADAPTIVE STREAMING

Yixuan Ban[1], *Lan Xie*[1], *Zhimin Xu*[1], *Xinggong Zhang*[1,2,*], *Zongming Guo*[1,2], *Yue Wang*[3]

[1]Institute of Computer Science & Technology, Peking University, Beijing, China
[2]Cooperative Medianet Innovation Center, Shanghai, China
[3]Beijing ByteDance Technology Co., Ltd.

banyixuan@bupt.edu.cn, {xielan, xuzhimin, zhangxg, guozongming}@pku.edu.cn,
wangyue.v@bytedance.com

ABSTRACT

To ensure 360-degree video's continuous playback and reduce the bandwidth waste, predicting user's future fixation is indispensable. However, existing methods concentrate either on user's motion information or content information. None of them consider users watching behaviors' inconsistency which embodies user's attention distribution more explicitly. So in this paper, we exploit Cross-Users Behaviors for viewport prediction in 360-degree video adaptive streaming, namely *CUB360*, trying to concurrently consider user's personalized information and cross-users behaviors information to predict future viewport. Besides, we use a QoE-driven framework to optimize existing video streaming approaches and propose a general algorithm aiming at solving the NP problem at a low complexity. Extensive experimental results over real datasets demonstrate that compared with traditional adaptive streaming method, our proposal can significantly boost the prediction accuracy by 20.2% absolutely and 48.1% relatively. Besides, the mean quality can get 30.28% gain while quality variance can be reduced by 29.89%.

Index Terms— 360-degree video, cross-users behaviors, viewport prediction, tile-based adaptive streaming, viewport adaptive streaming

1. INTRODUCTION

360-degree Video on Demand (360-VoD) is becoming increasingly popular on both industry and academia. It can create a completely immersive experience, allowing users to control their visual fixation freely during video playback through Head-Mounted Displays (HMDs). Traditional streaming platforms, such as YouTube, deliver the full projected panorama videos to clients [1]. This approach however is neither bandwidth efficient nor high-quality since only a portion of the entire video frame is viewed by the user. Recently, *tile-based viewport adaptive streaming* is regarded as a promising way to deliver 360-degree video efficiently [2–4]. Instead of transmitting the entire video frame, it delivers high quality tiles within user's viewport, while others are in low quality or even discarded. In addition, to ensure continuous playback, the client must keep a relative large buffer, which makes proactively pre-fetching tiles by viewport prediction necessary as well.

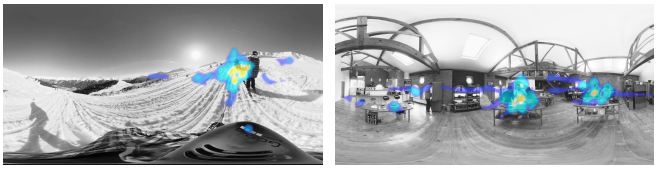


Fig. 1. User's visual fixation heat map on video frames.

Existing viewport prediction methods can be classified into two categories: *motion-based* [2, 3] and *content-based* [5]. The *motion-based* methods extrapolate user's future viewport according to user's history motion information from HMD – in terms of Euler Angle, Quaternion or Rotation Matrix. However, the long-term viewport prediction is highly inaccurate, e.g. for viewport after 2 seconds, the motion-based prediction accuracy could drop below 70% [2]. And the accuracy decreases rapidly along with prediction period. As for the *content-based* method, it considers the visual saliency of video to predict the viewport with the cost of high computational complexity. Besides, it exists bias since users may have different Region of Interest (ROI) and the correlation between user's watching behaviors and saliency information is unclear.

Unlike the aforementioned methods, we predict viewport leveraging cross-users viewing behaviors as guidance, which is inspired by user behavior's statistics. Intuitively, if we

*Corresponding author. E-mail:zhangxg@pku.edu.cn

This work was supported by National Natural Science Foundation of China under contract No. 61471009, Beijing Culture Development Funding under Grant No.2016-288.

978-1-5386-1737-3/18/$31.00

know most individuals are interested in one object within the picture, then we have good reason to believe other viewers also tend to gaze the same area. To verify this, we depict the position of all users' visual fixation through analyzing a real head-movement trace dataset [6]. As shown in Fig. 1, the visualization heat map includes 48 users' visual fixation during 69-70 sec of Freestyle Skiing (left) and 33-34 sec of Cooking Battle (right). The result indicates: 1) humans share similar watching behaviors, 2) the interesting area for users can be multiple, which is ubiquitous in 360-degree videos [6].

Inspired by these observations, we exploit Cross-Users Behaviors to predict viewport for 360 video adaptive streaming, called *CUB360*. In viewport prediction, our proposal combines the benefits of motion-based prediction and cross-users behaviors. From historical trajectories of head movements, a motion-based fixation is firstly predicted using Linear Regression (LR). Then, cross-users viewing fixations are exploited by K-Nearest-Neighbors(KNN) algorithm. K-nearst fixations around LR result are found to improve the prediction accuracy. Additionally, a QoE-driven rate allocation algorithm is proposed to optimize the rates of each tile within predicted viewport.

To summarize, our main contributions include:

- We propose a KNN-based Viewport Prediction approach called *KVP*, trying to concurrently consider personalized behavior and cross-users behaviors to predict user's viewport especially for long-term.
- We mathematically formulate the rate adaptation into an QoE-driven optimization framework and propose an Optimal Rate Allocation algorithm, *ORA*, to solve this problem.
- Extensive simulation experiments carried out on realistic datasets validate that *CUB360* can revolutionarily achieve 20.2% absolutely and 48.1% relatively gain on viewport prediction accuracy. Besides, the video quality can be improved by 30.28%.

The rest of this paper is organized as follows. Section 2 surveys related works on tile-based viewport adaptive video streaming and viewport prediction. In Section 3, we specifically describe the viewport prediction approach, *KVP*. The optimal streaming method and the general solution *ORA* are presented in Section 4. In the end, we discuss results of experimental performance in Section 5 and conclude this paper in Section 6.

2. RELATED WORK

Recently, viewport adaptive streaming is emerging as a promising way to deliver 360-degree videos. Among them, tile-based viewport adaptive streaming is regarded as a hot research direction which can offer spatial random access on 360-degree content. It crops original 360-degree video frames into multiple tiles and encodes them into multiple bitrate versions by modifying coding settings such as resolution, quantization parameter, etc.

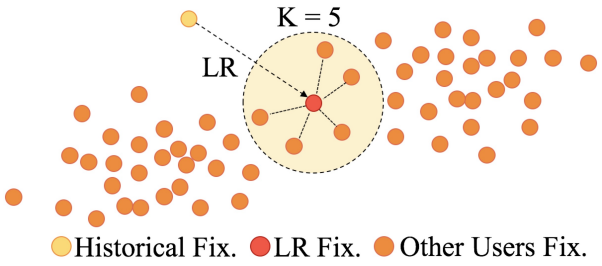


Fig. 2. KNN-based viewport prediction

As the key role in viewport adaptive streaming, viewport prediction is of crucial importance. However, the existing prediction methods only work for short-term (<2s) [2], which is not sufficient to ensure 360 videos continuous playback and may lead to blank block even stall. In general, the prediction approaches broadly fall into two major categories: *motion-based* and *content-based*.

Feng *et al.* [2] apply LR to predict the future fixations represented by Euler Angle (pitch, yaw and roll) separately. However, this naive approach's accuracy drops quickly especially for long-term, which could result in huge blank block. Lan *et al.* [3] perform a viewport probabilistic model considering the Gaussian Distribution of LR prediction error, which can not ensure accuracy for long-term neither. Both motion-based methods above are absence of content and behavior information inherently, so the gain is extremely limited.

As for content-based method, [5] builds a fixation prediction network which uses pre-trained CNN to generate saliency map. However, the computational complexity is relatively high since the neural network works pixel-wise. Besides, the correlation between saliency map and user's viewing likelihood has not been fully investigated, the ROI could also be multiple, which makes the performance not reliable enough.

3. KNN-BASED VIEWPORT PREDICTION

In this section, we formally present our KNN-based Viewport Prediction algorithm, *KVP*, which incorporates LR result and cross-users behaviors into design aiming at promoting viewport prediction accuracy especially for long-term. To calculate each tile's viewing probability and coordinate personalized and cross-users behaviors information reasonably, we partition KVP's implementation into three main steps including: a) *LR* (according to single viewer's history fixation), b) *KNN-Based Election* and c) *Probability Vote Mechanism*.

Specifically, as shown in Fig. 2, LR extrapolates user's possible fixation based on historical trajectories while KNN-based election picks the nearest K fixations of other users to amend the prediction result. Finally, the probability vote

mechanism is carried out to calculate each tile's viewing probability.

3.1. Linear Regression

Assuming viewer's fixation is represented as $O(\alpha, \beta, \gamma)$ in Euler angle, while α, β, γ representing yaw, pitch, roll, respectively. The same as [2], in system time t_0, we train a LR model in window $(t_0 - 1, t_0]$ using historical fixations to predict viewport in future δ time. The regression coefficient over yaw, pitch, roll is b_α, b_β, b_γ estimated by Least Squares Method (LSM), then the estimated fixation O_r generated by LR can be formulated as:

$$\begin{cases} \alpha_r(t_0+\delta) = b_\alpha \delta + \alpha(t_0), \\ \beta_r(t_0+\delta) = b_\beta \delta + \beta(t_0), \\ \gamma_r(t_0+\delta) = b_\gamma \delta + \gamma(t_0). \end{cases} \tag{1}$$

3.2. KNN-Based Election

Due to the observations on cross-users behaviors inconsistency, we adopt a KNN-based election method aiming at electing nearest K fixations O_f to amend the inaccurate LR result O_r. Since all the fixations are distributed on the sphere, the distance calculation should be conducted on sphere. Assuming the Euler angle is in the left-handed coordinate system adopted by the Unity 3D platform [6], in which the positive x, y and z axes representing the pitch, yaw and roll while pointing right, up and forward respectively. Then all the fixations can be transformed into unit directional vector in Cartesian coordinate system as below:

$$\begin{cases} x = \sin(\alpha) \cdot \cos(\beta) \\ y = \sin(\beta) \\ z = \cos(\alpha) \cdot \cos(\beta) \end{cases} \tag{2}$$

Ultimately, the sphere distance between LR result and other users fixations $D(O_r, O_f)$ can be calculated as:

$$D(O_r, O_f) = \arccos(x_r \cdot x_f + y_r \cdot y_f + z_r \cdot z_f) \tag{3}$$

Based on the sphere distance, we can sort and number the nearest K fixations as O_f^k from smallest to largest representing cross-users behaviors, where $k \in \{1...K\}$.

3.3. Probability Vote Mechanism

Given LR result and the nearest K fixations, to generate a series of tile probability for subsequent streaming, we propose a vote mechanism considering fixation weights and tile's viewed times. Firstly, LR's accuracy decreases rapidly as prediction time extends, so its weight should be diminished by time. We denote it as $w_r = \frac{1}{\delta}$. As for other fixations, we distribute a constant weight $w_f = 1$.

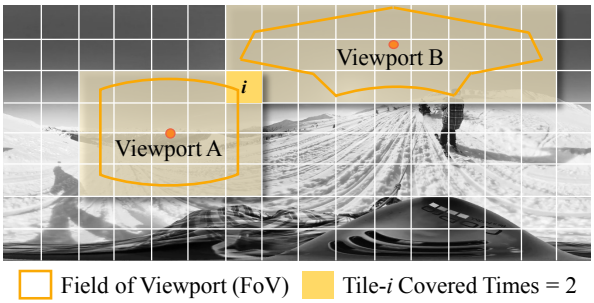


Fig. 3. Probability vote mechanism

Secondly, as shown in Fig. 3, each fixation actually corresponds a covering area on the plane. For each tile, the covered times should be proportional to the viewing probability. Assuming $\boldsymbol{L}(O)$ is the covering fields of each fixation represented by a N-dimension vector, where N represents the tile number indexed in raster-scan order. $L_i = 1$ means the tile is viewed while $L_i = 0$ otherwise. Then for certain tile, the votes V_i can be inferred as below:

$$\begin{aligned} V_i &= w_r \cdot L_i(O_r) + \sum_{k=1}^{K} w_f \cdot L_i(O_f^k) \\ &= \frac{1}{\delta} \cdot L_i(O_r) + \sum_{k=1}^{K} L_i(O_f^k) \end{aligned} \tag{4}$$

To depict the distribution of each tile's viewing probability p_i, we normalize votes $\boldsymbol{V}$ as below:

$$p_i = \frac{V_i}{\sum_{i=1}^{N} V_i} \tag{5}$$

4. OPTIMAL VIDEO STREAMING

In this section, we present a QoE-driven tile-based adaptive streaming system which leverages video's quality optimization framework to maximize user's quality while minimizing spatial quality variance and rebuffering. In the meanwhile, we propose a general method called Optimal Rate Allocation, *ORA*, trying to find the optimal solution under this NP problem.

4.1. Optimization Function

In tile-based adaptive streaming, the original 360-degree videos are spatially cropped into N tiles with raster-scan order. Then each tile is temporally divided into several continuous segments with fixed duration T and encoded by M bitrate levels. Consequently, there are $N \times M$ kinds of tiles in one segment waiting to be delivered on the server side. To derive the optimal tile sets, we denote $j \in \{1...M\}$ as the bitrate level while letting $r_{i,j}$ and $d_{i,j}$ denote the bitrate and distortion of tile i in j-th rate. Besides, we denote $\boldsymbol{X} = \{x_{i,j}\}$ as

the choosing results, where $x_{i,j} = 1$ means the tile is selected and $x_{i,j} = 0$ otherwise.

To maximize user's QoE, the same as [3], we define two QoE functions $\Phi(\boldsymbol{X})$ and $\Psi(\boldsymbol{X})$ representing video's expected quality distortion and quality variance separately. Then for the purpose of minimizing both quality distortion and variance, our optimization problem can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{X}} \quad & \Phi(\boldsymbol{X}) + \eta \cdot \Psi(\boldsymbol{X}) \\ \text{s.t.} \quad & \sum_{i=1}^{N}\sum_{j=1}^{M} x_{i,j} \cdot r_{i,j} \le R, \\ & \sum_{j=1}^{M} x_{i,j} \le 1, \qquad x_{i,j} \in \{0,1\}, \qquad \forall i. \end{aligned} \tag{6}$$

Specifically, η represents the weight for quality variance while R representing the available bandwidth generated by rate adaptation algorithm, which is used to reduce rebuffering. The other limitation ensures each tile should be delivered in one rate in case of unnecessary waste. The same as [3], the expected quality distortion can be formulated as:

$$\Phi(\boldsymbol{X}) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M} D_{i,j} \cdot x_{i,j} \cdot p_i}{\sum_{i=1}^{N}\sum_{j=1}^{M} x_{i,j} \cdot s_i}, \tag{7}$$

Note that p_i is derived from our proposed KVP algorithm while s_i and $D_{i,j}$ representing each tile's spherical area and spherical distortion. Numerically, the relationship obeys:

$$D_{i,j} = d_{i,j} \cdot s_i. \tag{8}$$

In tile-based streaming, the spatial quality variance is strictly related to the seams between adjacent tiles rather than the overall quality variance. Besides, calculating all tiles variance introduces much computational burden for the system. So in this paper, we proposed a novel lightweight method to depict the spatial quality variance. Contrary to [3], we denote the tile number assemblage adjacent to tile i as U_i, then the $\Psi(\boldsymbol{X})$ can be formulated as:

$$\Psi(\boldsymbol{X}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{u\in U_i} p_i \cdot p_u \left|\sum_{j=1}^{M}(x_{i,j} \cdot D_{i,j} - x_{u,j} \cdot D_{u,j})\right| \tag{9}$$

4.2. ORA Algorithm

Since the QoE-driven framework's objective is to find an optimal solution under limited resources, the *ORA* algorithm can be regarded as a knapsack problem, which is a typical NP complete problem. Before describing the solving process concretely, we firstly introduce the definition OPT_r, which is the optimization function value corresponding to the optimal selection $\boldsymbol{X}^r$ under available bandwidth $r \in \{1...R\}$.

Algorithm 1 Optimal Rate Allocation (ORA) Algorithm

Input: (1) Tile's probability p_i. (2) Available bitrate R.
Output: $\boldsymbol{X}^R = \{x^R_{i,j}\}$.

```
1:  /* Function 1: ORA main function */
2:  Set OPT_r to be MAX and initialize X^r = 0;
3:  for i ∈ N do
4:      for r ∈ R do
5:          for j ∈ M do
6:              if r > r_{i,j} then
7:                  OPT_r = min{OPT_r, Com(i, j, r)};
8:          Update(X^r);
9:  return X^R = {x^R_{i,j}};
10: /* Function 2: Recompute OPT_r adding x_{i,j}^{r-r_{i,j}} = 1 */
11: Com(i, j, r){
12:     x_{i,j}^{r-r_{i,j}} = 1;
13:     OPT_r = Φ(X^{r-r_{i,j}}) + η · Ψ(X^{r-r_{i,j}});
14:     x_{i,j}^{r-r_{i,j}} = 0;
15:     return OPT_r;}
```

As shown in Algorithm 1, each tile's viewing probability p_i and available bandwidth R are our proposal's input while the best selection result $\boldsymbol{X}^R$ is the output. To minimize the QoE function, we firstly set all the OPT_r as a big enough number *MAX* while initializing $\boldsymbol{X}^r$ as zero (line 2).

For certain tile i (line 3), under certain bandwidth r (line 4), in certain bitrate j (line 5), there could be two choices, selecting this tile or not. In both cases, we have to recompute OPT_r by adding $x_{i,j}^{r-r_{i,j}} = 1$ to the optimal selection $\boldsymbol{X}^{r-r_{i,j}}$ (line 12). Then tile i should be selected only if the new OPT_r is smaller (line 7). To record it, for each rate, we have to update the selection $\boldsymbol{X}^r$ (line 8). One thing should be noticed that in function $Com()$, we only recompute the OPT_r based on new adding tile, when the function exits, the $\boldsymbol{X}^r$ get back to the old one in case of error recording (line 14). After this, we can obtain the best selection over rate R, which is $\boldsymbol{X}^R$ (line 9).

5. PERFORMANCE EVALUATION

5.1. Experimental Configuration

In our experiments, we evaluate the strengths of CUB360 under simulated environments. Specifically, we use realistic HSDPA bandwidth dataset [7] and Wu's trajectory dataset [6] including 18 videos and 48 users. To meet practical requirements, the same as [3], we partition each video into 6×12 tiles ($N = 72$) and crop them into segments every second ($T = 1$). Then all the tiles are encoded into 5 different bitrates ($M = 5$) including {20kbps, 50kbps, 100kbps, 200kbps, 300kbps} using open source encoder x264. After that, each tile's planar distortion $d_{i,j}$ can be calculated by comparing them with raw panoramic videos.

Table 1. Video Information and Viewport Deviation

Video ID	Content	Category	LR (%)	CUB360 (%)
Video-1	Conan360°-Sandwich	Performance	42.2	**14.1**
Video-2	Freestyle Skiing	Sport	38.3	**27.5**
Video-3	Google Spotlight-HELP	Film	43.8	**22.1**

As discussed above, the accuracy of viewport prediction is closely associated with prediction time. In practice, the prediction duration is equal to buffer size actually. Based on that, we apply the target-buffer based rate adaptation algorithm [3], which can adjust downloading bitrates to make sure smooth play. To demonstrate our proposal's improvement on viewport prediction especially for long-term, we set the buffer size into {2s, 3s, 4s, 5s, 6s} separately for comparison.

To explore our proposal's performance, we compare CUB360 with the representative work [2] which leverages LR as prediction method and delivers predicted tiles at the same rates. In system evaluation, we take the following metrics into consideration:

- Viewport Deviation: It calculates the percentage of black area rendering over the screen which could damage user's experience, which implies the distribution of prediction accuracy.
- Viewport PSNR: It uses the Peak Signal to Noise Ratio (PSNR) in viewport to effectively depict video quality and user's experience, which is widely adopted in research fields [8].
- Viewport Quality Variance: It adopts Coefficient of variation (CV) to evaluate the viewport quality variance in one segment.
- Bandwidth Occupation: It calculates the total bitrates for each segment, which reflects the ability of utilizing data and video quality.

5.2. Viewport Deviation Performance

As shown in Table 1, to demonstrate our approach's generality, we pick three typical videos in [6] covering different categories including performance, sport and film as comparison. To eliminate the impact of users' individual difference, we denote specific user's trajectory as viewer while others as reference in turns. Besides, we change the buffer size and video ID in turns to observe the performance under different conditions.

As shown in Fig.4, as buffer size increases, both LR and CUB360's black ratio increases. However, our proposed method CUB360's increase is not so drastic as LR especially in video-1 (Fig.4(a)), which is because under performance video, user's attention is more concentrated. Besides, CUB360's medians of the three videos all lie next to zero basically, which once again proves our proposal's high prediction accuracy. Numerically, as shown in Table 1, when buffer size is 6s, implemented by CUB360, the *absolute improvement* of these three videos can reach up to 28.1%, 10.8% and 21.7% separately, 20.2% on average. The average *relative improvement* can even obtain 48.1%.

5.3. Video Quality Performance

The video quality performance is illustrated in Fig. 5 and the average performance is listed in Table 2. To reveal our proposal's property, we set the target-buffer as 6s to compare the video quality performance. As shown in Fig.5(a), for each video, the two curves of viewport PSNR both follow the same trend, however, the difference in quantity reveals our proposal's outperformance. Especially in video-3, almost 50% of the viewport PSNR are over 40 dB in CUB360, as for LR, the percentage declines sharply to 18%. Overall, the average improvement can reach up to 30.28%.

As for viewport quality variance, as shown in Fig.5(b), once the black ratio is over 90%, the quality variance is unbearable in a sense, so we set a threshold at this point denoting the corresponding quality variance here as 10, which is reasonable for these experiments. Then, as shown in Table 2, CUB360 can get average 29.89% gain on quality variance. Besides, from the percentage on the maximum, 10, we can infer that the LR is more likely to suffer from blank block.

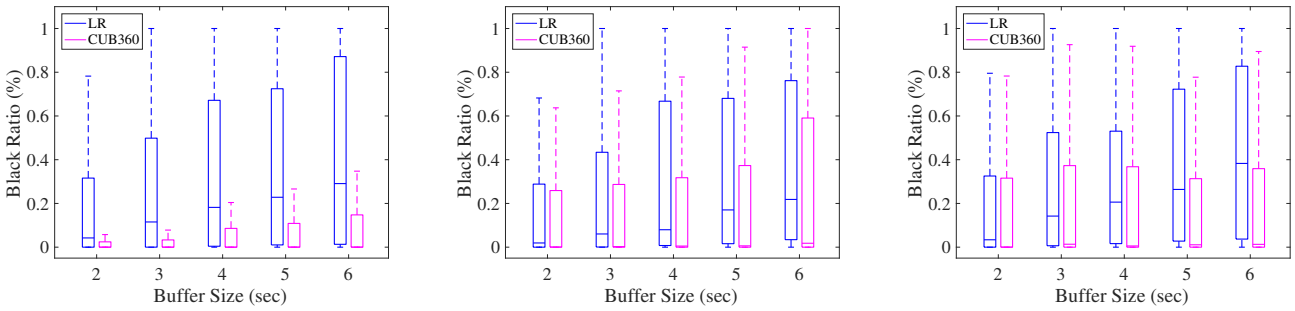
In fig.5(c), we can observe that CUB360's bandwidth occupation greatly exceed LR, almost attains 33% on average. Besides, the peculiar sawtooth of LR's curves caused by quantization implies LR's low utilization on bandwidth, which is absence on CUB360.

Table 2. Average Quality Performance and Improvement

Metrics	LR	CUB360	Relative Imp.
Viewport PSNR (dB)	25.76	**33.56**	+30.28%
Viewport Quality Variance (CV)	3.78	**2.65**	+29.89%
Bandwidth Occupation (kbps)	1570.80	**2077.53**	+32.26%

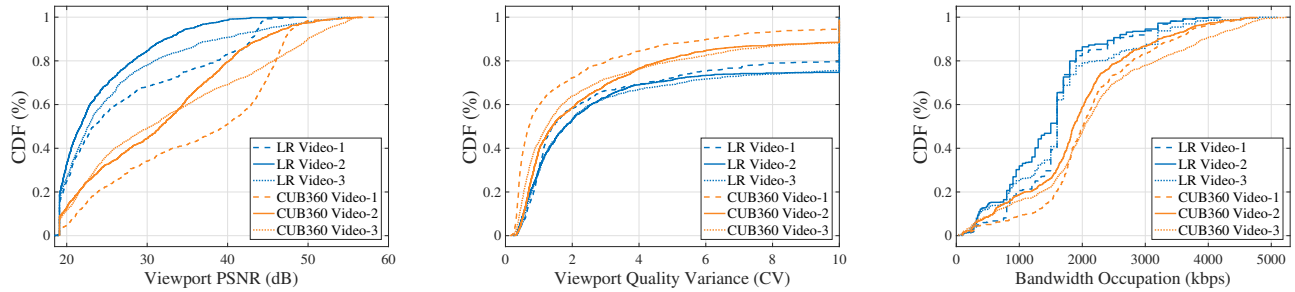
6. CONCLUSION

In this paper, we propose a novel framework CUB360 which leverages cross-users behaviors to predict viewer's fixation especially for long-term and utilize the QoE-driven optimization problem to realize optimal video streaming. Besides, we present a general approach *ORA* to solve the range of rate allocation problems. Compared with traditional LR algorithm, our proposal CUB360's viewport deviation can be improved by 20.2% absolutely and 48.1% relatively. The improvement of viewport PSNR, quality variance and bandwidth can reach up to 30.28%, 29.89% and 32.26% separately.



(a) Viewport Deviation of Video-1 (b) Viewport Deviation of Video-2 (c) Viewport Deviation of Video-3

Fig. 4. Viewport Deviation results



(a) CDF of Viewport PSNR (b) CDF of Viewport Quality Variance (c) CDF of Bandwidth Occupation

Fig. 5. Video Quality Results

7. REFERENCES

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

[2] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proc. ACM SIGCOMM AllThingsCellular*, 2016, pp. 1–6.

[3] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in *Proc. ACM Multimedia*, 2017, pp. 315–323.

[4] M. Hosseini and V. Swaminathan, "Adaptive 360 vr video streaming based on mpeg-dash srd," in *Proc. Multimedia (ISM), 2016 IEEE International Symposium on*, 2016, pp. 407–408.

[5] C.-L. Fan, J. Lee, W.-C. Lo, C.-Y. Huang, K.-T. Hsu, and C.-H. Hsu, "Fixation prediction for 360 video streaming in head-mounted virtual reality," in *Proc. ACM NOSSDAV*, 2017, pp. 67–72.

[6] C. Wu, Z. Tan, Z. Wang, and S. Yang, "A dataset for exploring user behaviors in vr spherical video streaming," in *Proc. ACM MMSys*, 2017, pp. 193–198.

[7] H. Riiser, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Dataset: Hsdpa-bandwidth logs for mobile http streaming scenarios," 2012.

[8] M. Yu, H. Lakshman, and B. Girod, "A framework to evaluate omnidirectional video coding schemes," in *Proc. IEEE ISMAR*, 2015, pp. 31–36.