Joint Rate and FoV adaptation in immersive video streaming

Dongbiao He
Tsinghua University
Beijing China
hdb13@mails.tsinghua.edu.cn

Cedric Westphal
University of California, Santa Cruz
Santa Cruz, CA
cedric@soe.ucsc.edu

J.J. Garcia-Luna-Aceves
University of California, Santa Cruz
Santa Cruz, CA
jj@soe.ucsc.edu

ABSTRACT

The responsiveness of the network is critical when the application is immersive video streaming or 360 degree video streaming. The users look at the video stream on a display that can only show a fraction of the full video stream, and the time that the information spend in transit dramatically impacts the QoE of the end user. Further, transmitting a 360 degree video stream significantly increases the bandwidth usage and the impact on the already strained network. We propose a mechanism to use the responsiveness of the network to perform a Field of View (FoV) adaptation, so as to reduce the bandwidth consumption while at the same time enhancing the QoE of the user. In our simulation, such mechanism significantly improves the resolution seen by the end user, by selecting a higher bit rate. This translates to an improvement of up to 1.34 × better resolution when compared with the full spherical 360 degree video stream. Further, when rate adaption is used, it ensures the QoE of end users evolves according to the changes in the network conditions. This means that our proposed rate and FoV adaptation based upon the network responsiveness achieves both the apprently contradictory goals of increasing the end-user QoE and reducing the overall bandwidth consumption at the network layer.

CCS CONCEPTS

• **Computing methodologies** → **Virtual reality**; • **Networks** → **Application layer protocols**;

KEYWORDS

Rate adaptation, Field of View, 360 degree video streaming, AR/VR

ACM Reference Format:
Dongbiao He, Cedric Westphal, and J.J. Garcia-Luna-Aceves. 2018. Joint Rate and FoV adaptation in immersive video streaming. In *VR/AR Network'18 '18: Morning Workshop on Virtual Reality and Augmented Reality Network, August 24, 2018, Budapest, Hungary.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3229625.3229630

1 INTRODUCTION

Augmented Reality and Virtual Reality are becoming common place. Facebook and YouTube have deployed support for some immersive videos, including 360 degree videos. Many companies have been offering devices to view virtual reality, ranging from simple mechanical additions to a smart phone, such as Google Cardboard to full fledged dedicated devices, such as the Oculus Rift. While the first commercial deployments, such as Google Glass for augmented reality, were inauspicious, this new wave of products promises an enhanced experience to users [3].

In such applications, the responsiveness of the network is critical. In particular, for immersive video streaming, the user will consume the information using a head mounted display, and the displayed content needs to be delivered in a timely manner. The amount of transmitted content increases dramatically with these applications.

Current networks still have a hard time delivering high quality video streams. In the future, 5G networks will have to address the challenges introduced by the new applications delivering augmented reality and virtual reality services. The 5G white paper [2] mentions augmented reality, 3D-video and pervasive video as use cases for dense urban networks. However, it is not clear it will be possible to deploy such applications without significant effort to reduce their bandwidth impact on the network.

AR/VR uses a set of tiles to display a Field of View (FoV) on the device's screen. The Field of view (FoV) is the open observable area a user can see on the display, and corresponds to only a fraction of the video stream. In typical displays, the FoV corresponds to roughly one sixth of the entire spherical view. Since the FoV is displayed to the user, and the rest of the video stream outside of the user's view is never consumed, the challenge is to transmit over the network only what is necessary.

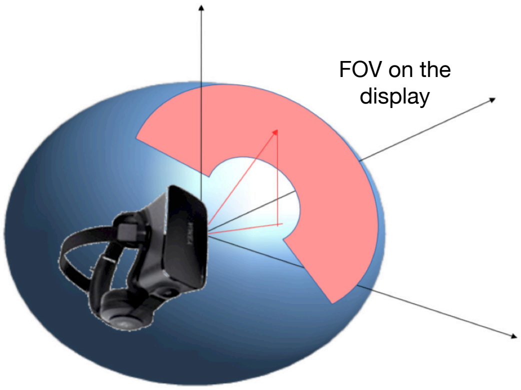


Figure 1: FoV

There is a basic trade-off between the responsiveness of the network, and the size of the FoV that the network must transmit. For illustration sake, assume that this view corresponds to a view angle of 90 degrees vertically and 110 degrees horizontally. We can compute what fractions of the sphere this corresponds to (say, roughly

VR/AR Network'18 '18, August 24, 2018, Budapest, Hungary

ACM ISBN 978-1-4503-5913-9/18/08.
https://doi.org/10.1145/3229625.3229630

1/6th). We use the measure of the network delay to compute FoV adaptation. We care about reacting to the end-to-end latency in the network. The survey [16] displays a number of prediction methods to estimate the network latency in different kinds of network. If the network delay is ideally small, the network is fully responsive and any requested data is transmitted immediately. In this case, the network only has to transmit the view that is displayed on the screen. On the opposite, if the network response is very long, say several seconds, then the client will request the data to be streamed, but will not know ahead of time where the user will be looking at the (much later) time it will be displayed. As the user can turn around within a few seconds, the whole spherical data stream needs to be downloaded on the device to anticipate the user's movements. Taking this into account, there is a bandwidth-delay trade-off depending on the responsiveness of the network perceived by the user, and the Quality of Experience rendered on the display.

Prior works such as [6] quantify the motion of the user as a function of time. In most cases, the users move less than 1 unit of distance within 1 second (where the distance is measured on the surface of the unit sphere, and therefore the maximal distance between two points is equal to π). Thus, we can categorize the network's responsiveness into different sizes of FoV that include the view within the expected distance within which the user's gaze may move during the time to retrieve and display the data. After analyzing the time delay of fetching a segment of 360 degree video, we put forward an algorithm which can offer the required FoV with high bitrate. Some work has focused on predicting the user behavior in order to identify the FoV that the user will be looking at in the future based upon the current gaze. We take a different –and complementary– approach and suggest here to adapt the field of view to the measured delay.

Namely, we contend that, when the network delay is close to zero, the client should only request a field of view that is slightly larger than the display on the screen (including a small margin around the sides of the current display view). We suggest that the requested FoV should increase as the network delay increases, until the whole sphere is transmitted for longer network delays. Once the proper FoV has been identified, the usual rate adaptation mechanisms should be invoked to ensure that the data will be received in a timely manner at the best possible resolution. Indeed, without rate adaptation, congestion in the network would increase the delay, which in turn would increase the FoV, and so on into a vicious cycle.

The main contribution of this paper is therefore to propose a rate and FoV adaptation mechanism which depends on the responsiveness of the network perceived by the user.

The paper is organized as follows: in Section 2 we discuss the related work. Section 3 discusses the relation between the play-back delay and the predictability. In Section 4 we present the network model and our rate and FoV adaptation algorithm. In Section 5, we describe the simulation environment and evaluate the performance of the alogrithm. Finally, we offer concluding remarks in Section 6.

2 RELATED WORK

360 degree video streaming is one of the key components of Virtual Reality (VR) applications. In the recent years, a number of solutions have been proposed to address the transmission of 360 degree video streaming. We summarize them into two categories:

Viewport based Transmission: 360 degree video is always divided into tiles. The tiles in FoV would have a high resolution for transmitting while the rest tiles might transmit in low quality. Qian et al. [17] looked at ways to minimize the amount of transmitted data with a prediction mechanism to identify what direction the user will be looking at. Using this prediction, only the predicted views are transmitted (plus some buffer around it to correct for prediction errors), thereby reducing the transmitted bandwidth. Xavier Corbillon et al. [7] put forward some theoretical models for the preparation of 360-degree video for viewport-adaptive streaming systems. And In [8], they improve the QoE of users by integrating the view navigation pattern and the spatiotemporal rate-distortion characteristics of the video content. Liu Xing et al. [13] use big data and crowd-sourcing to add intelligence to streaming algorithms for predicting the view of users. Fanyi Duanmu et al. [9] propose a novel two-tier dynamic 360 degree video streaming framework, which encodes 360 degree video into two tiers and adaptively streams the two tiers of the video to cope with the variations in the network bandwidth and user view direction changes. Tarek [11] showed how a tiled streaming framework, where a degrading quality model similar to the state-of-the-art offset-cubemap is used, minimizes the storage requirements at the server side and improves the end user's QoE. Mohammad Hosseini et al. [12] proposed a modification of DASH to support tiled streaming. F. Duanmu et al. [10] as well considered the benefits of tiled streaming for 360 degree videos.

Caching and Edge Computing: Edge computing is expected to be an effective solution to deliver 360-degree virtual reality (VR) videos over networks. Cache is one of underlying resources for enabling these applications. Mangiante et al. [14] present a FoV rendering solution at the edge of a mobile network, designed to optimize the bandwidth and latency required by VR 360 degree video streaming. Jacob Chakareski et al. [4] designed an optimization framework that allows the base stations to select cooperative caching/rendering/streaming strategies that maximize the aggregate reward they earn when serving the users, for the given caching/computational resources at each base station. Zhang Liyang et al. [19] proposed a VR video conferencing system over named data networks (NDN), which is a in-network caching supported architecture. In [15], they propose a low latency and low loss streaming mechanism called L4C2 via in-network coding and caching mechanism, which is more convenient for high-quality real-time delay-sensitive streaming.

In addition, Afzal et al. [1] considered the properties of 360 degree video streaming through a study of YouTube videos and made a comparison with a regular video. [18] considered the challenges created upon the network by immersive applications such as AR/VR and suggested to consider Information-Centric Networking as a possible architectural answer to these challenges.

3 DELAY AND PREDICTABILITY

In order to provide satisfying QoE for an end user, we should avoid the re-buffering events between segments. These are created by two reasons: (1) the delay of network; (2) the mismatch between the requested FoV and what FoVs have been stored in the playback

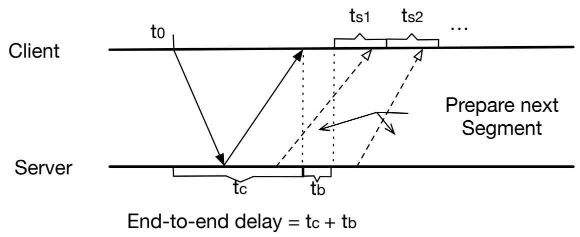


Figure 2: Overview of Delay

	100ms	250ms	500ms	750ms	1000ms
0.95	0.147	0.433	3.012	3.093	3.107
0.90	0.096	0.255	0.567	1.11	2.983
0.85	0.073	0.19	0.401	0.645	0.956

Table 1: Distance variance with increasing of time

buffer. Hence, we need to transmit a larger FoV than the display size, in order to accommodate for the motion of the user's gaze within the 360 degree video stream during the time between the request for the data, and the time at which this data is displayed on the screen. First of all, we need to estimate how much distance this gaze will travel during that time period.

The distance is measured on the surface of the unit sphere, where the user is placed at the center. We measure the distance covered by the center of the FoV that the user is looking at. Because this distance is on the unit sphere, it varies between 0 and π.

In practice, for a streaming application, the time to retrieve and display a segment has to include several components:

- the user will request at time t_0 a segment that will lasts t_s seconds of playback time;
- this segment will be retrieved from the server after a network transmission completion time t_c at $t_0 + t_c$.
- This segment will then be buffered into a playback buffer, and played back after a buffer delay t_b.
- Finally, the segment will start playing at time $t_0 + t_c + t_b$ and conclude at time $t_0 + t_c + t_b + t_s$.

Figure 2 illustrates the whole procedure for the clients to retrieve the required video segment. The network delay would play a important role in choosing the length of segment. The paper [8] claims that a segment length t_s of 2s offers the right amount of predictive power, namely that the center of the FoV will stay within a reasonable distance of the current center within the next 2s. Note that this is the distance from the time the user starts watching the segment, as a function of the segment length. Namely, each segment starts with a given FoV and this measures the probability that all throughout the segment length t_s, the user's gaze stays within some reasonable distance of the initial center, per [8]. Therefore it only includes the time t_s in their analysis. The rest of the delay between the request and the start of the playback is not included. However, it is at the request time t_0 that the FoV needs to be selected that will be played at time $t_0 + t_c + t_b + t_s$.

Using the datasets provided by [6] , we are able to generate the figure and table below (See figure 3 and table 1) that shows how far the user's FoV moves during a given time interval. For instance, we can see that there is a 90% probability that the user will be viewing a FoV that is within distance 0.096 at time 100ms from the initial FoV. (The distance is defined as maximum distance of the shortest path on the sphere between the center of the FoVs within the time interval.) When we enlarge the time interval to 750*ms*, the distance of the users changes is within 1.11 with 90% probability. Even with the 1*s* delay, there is a 85% chances that the user stays within distance of 0.956 from the initial position. Hence, with the estimated network delay, we can use table 1 to adapt the FoV.

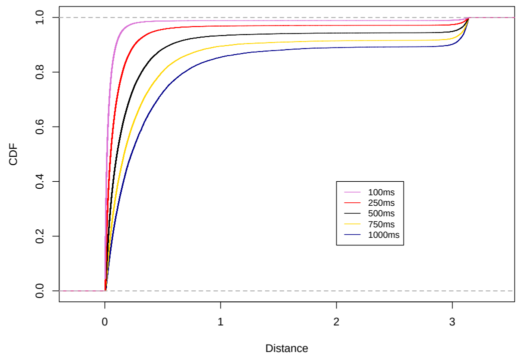


Figure 3: Distance from Initial FoV as a Function of Time

4 MODEL AND PROPOSED FOV ADAPTATION MECHANISM

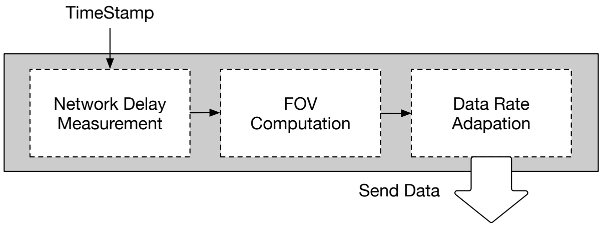


Figure 4: The procedures of Joint FoV and Rate Adaptation Mechanism

We propose to adapt the FoV based upon the measured delay and bandwidth in the network. Namely, after obtaining the real time network delay, we can specify a playback buffer time τ, so that the next segment can be downloaded in time to be played. For convenience, the network delay is estimated from the observed network Rount Trip Time (RTT), denoted as t_n, in this paper. The whole procedure of this algorithm is showed in figure 4.

The minimum completion time for a segment S of size s can be approximated by: $t_c(S) = t_n + \frac{s}{C}$ where C is the link capacity. For the packet to be delivered on time, we need to satisfy:

$$t_c(S) = t_n + \frac{s}{C} \leq \tau \tag{1}$$

where $t_c(S)$ is equal to the network completion time we illustrated in section 3 (See figure 2) and t_n is the RTT. Initially, if a packet is requested at time 0, it will start playing at time τ for τ units of time. This entails that the FoV size should be dimensioned such as at time 2τ, the probability that the user will stay within some distance d of the requested FoV is over, say, 90%. Based upon the previous section, we can look up the relationship between τ and d, such that the request includes all the area within distance d of the requested FoV.

For a specific FoV size, we can then compute the resolution such that the size s of the packet satisfied, namely pick the highest rate (link capacity) such that:

$$s \leq C \times (\tau - t_n) \tag{2}$$

τ defines the choice of the FoV from the equation. And s depends on the resolution of the packet and the distance of FoV. For simplicity, we can assume that the selected FoV is transmitted and the rest of the stream is not. However, it is a very simple change of the algorithm to transmit the selected FoV at a higher resolution, and the rest of the video stream at a lower resolution. For instance, the algorithm could be modified very simply to transmit a full spherical stream with a base layer resolution, and the selected FoV with an enhanced layer resolution. For formulation of the allocation on the segment of s, we define p_i as the time that the user would gaze on the i^{th} tile (here we assume that the total number of tiles is n), and x_i as the chosen encoded bitrate. Besides, a boolean parameter w_i is introduced to indicates the selection of tiles in the segment. Then, we can write the maximum gain for allocating the segment as:

$$\max \quad \sum_i^n p_i x_i \tag{LP1}$$

$$\text{s.t.} \quad \sum_i^n x_i * w_i \times (\tau - t_n) \leq s, \quad w_i \in \{0, 1\} \tag{3}$$

$$p_i x_i > 0, \qquad \forall p_i > 0 \tag{4}$$

From the formulation above, equation 3 represents that the sum of encoding bitrate could not exceed the segment size we obtained in equation 2. Apart from the size restriction, p_i is another major factor in choosing the quality of transmission for each tile. Namely, the tiles in FoV that have a high probability of being seen should be downloaded at high resolution, and the rest at low resolution. We can use the statistical strategy mentioned in [5] for assigning the value of p_j, which is also a weight of each tile called navigation likelihood. We can also adapt the value of p_i for the tiles outside of FoV when there is an extra segment of s left or reserved.

Since we use the current end to end network delay to measure the adaptation, it will not be efficient when the network state changes greatly with the fluctuation of the sending rate. Besides, for a better QoE, the transmission should start with a lower quality and improve over time. Taking into account these factors, we introduce two parameters R_{low} and R_{high} into this algorithm. R_{low} is the threshold which represents that we can send the 360 degree video at the best quality without much delay, and there is no need to worry about the changes of network. We can keep using the maximum sending rate when the delay is smaller than R_{low}. R_{high} is the threshold that the quality of experiences will be affected greatly by the network state. We should decrease the sending rate with a decrement factor α if the delay is larger than R_{high}.When the delay is between R_{low} and R_{high}, the increment of sending rate is based on the previous sending rate by a factor β. The major goal of this rating adaption algorithm is to place the sending rate between R_{low} and R_{high} which makes the prediction of FoV with network delay effective.

Algorithm 1: Algorithm of joint rate and FoV adaptation

```
1  INPUT: $new\_delay, rate_{pre}$;
2  OUTPUT: $FoV$ and segment size $s$;
3  if $findthecorrespondingdelayinTable1$ then
4      $\tau = f'(new\_delay)$;
5      $FoV = FoV\_base + 2 * FoV\_dis(i + 1)$;
6  end
7  $C$=get($link\_rate$);
8  if $new\_delay < R_{low}$ then
9      $rate = Max\_rate$;
10 end
11 if $new\_delay > R_{high}$ then
12     $rate = rate_{pre} \times (1 - \alpha.(\frac{R_{high}}{new\_delay}))$;
13                                         ▷ $\alpha$ : a decrement factor
14 end
15 if $R_{low} \leq new\_delay \leq R_{high}$ then
16     $rate = rate_{pre} + \beta$;
17                                         ▷ $\beta$ : additive increment step
18 end
19 if $rate > C$ then
20     $rate = C$;
21 end
22 $s = rate * (\tau - t_n)$;
23 Return $FoV$, $s$;
```

The proposed algorithm for joint rate and FoV adaptation is therefore(see Algorithm 1):

- Measure t_n(RTT) over a period of time;
- Define τ such that $\tau > \max_{t-k,t}(t_n) + \frac{s}{C}$, where k is a window over which to estimate the variations of t_n(line 4);
- Find the FoV distance d such that all views are within distance d of the initial position within 2τ (line 3-6);
- For a period of k units of time, transmit the area within distance d at high resolution, while satisfying Equation 2(line 5);
- Update sending rate with the threshold of R_{low} and R_{high} in case of the congestion when taking full use of the link capacity (line 8-21);
- Update Segment size s based upon the measurement from this period; return the FoV size accordingly (line 22,23).

5 PERFORMANCE EVALUATION

We built a simulator based on NS-3 to evaluate the performance of the joint rate and FoV adaptation. During the simulation, we assume the base horizontal and vertical FoV angles are 110° and 90°, and we use the distance value to measure the view we need to fetch. We use four static settings of FoV for the comparison of our design, namely the distance of 1.0 (D1), 1.35 (D1.35), 1.5 (D1.5) and π (DF). Once the simulation starts, our design will calculate the FoV according to the value of network delay, and get the best bitrate quality that matches the network conditions per the algorithm from the previous section.

This evaluation demonstrates that:

- The algorithm of joint adaptation of FoV and date rate supports a 1.34× improvement in resolution when compared with transmitting the full spherical view; it also requires only 94.5% of the average bitrate required to transmit only the base view to the users. Namely, it achieves a better resolution with a lower bandwidth.
- The algorithm achieves the best QoE score compared with the other transmitting strategies since it makes a connection with the time delay and the downloading segment when fetching the views.

5.1 The network delay and Bitrate

The line graph of Figure 5(a) demonstrates the round trip time of the different settings of FoV over a short period. It is clear that the network delay varies with the five strategies. However, the trends for AF, D1.35, D1.5 and DF are very similar, and they would step into a periodic variation around 3 seconds after the simulation started. D1 will have a static delay value which is equal to $356ms$ after the time point of 3.586. From the figure, we can find that AF could achieve the minimum delay, which is as low as 48, while DF has the maximum value with a long variation period since it needs to transmit the whole view of the screen.

Figure 5(b) compares the bitrate among the five strategies tested in this part. In this simulation, D1 could achieve the largest bitrate ($1547kb/s$), since it transmits the minimum size of FoV, which is an ideal situation. However, we can see than the gap between AF and D1 is not large since AF will always change the size of view according to the variation of network delay. DF could not adapt its bitrate to catch up with other four strategies within the 100 seconds. From this figure, we can also discover that if we transmit a smaller FoV, we can achieve a higher bitrate with a shorter time. For example, D1 achieves $791kbps$ in 26 second while AF needs extra 8 seconds to achieve the same bitrate. As time grows, the gap becomes larger. From this figure, D1 would be an ideal situation without the concerns of QoE which we will discuss later. Figure 5(c) evaluates the average bitrate of these cases. D1 has the highest average bitrate ($550kbps$) while AF could achieve $520kpbs$. The difference between them is only $30kbps$, which illustrates that AF could have a good quality of video transmission rate.

The bar charts of Figure 6(a) and Figure 6(b) calculate the ratio of different size of FoV in two kinds of link rate. We can see that we need to frequently transmit all view of the sphere for the users when the link rate is low. In figure 6(a), the FoV of distance 3.14(full sphere) takes almost 82% among all the views. However, when the bandwidth increases to $1000kbps$, the ratio of 3.14 dropped to 0 since there is no need to transmit all the view with the permit of the network state(showed in figure 6(b)). Interestedly, FoV of 1.15 and 2.10 take the 50% of the views each in our experiments. Hence, if the bandwidth is low, our design will reduce to the situation that the full sphere should be transmitted to users. On the contrary, we transmit the short distance to the clients when the network has a small delay in order to increase the quality of experience for users.

5.2 QoE score

In this part, we start to evaluate the QoE score of the five cases. We use the dataset in the [6] as the user traces for getting the requirements of the views and latency in real time. Then we can obtain the QoE score according to the following equation:

$$QoE\_score \propto \frac{Bitrate}{freq_{miss} \times Delay} \tag{5}$$

where $freq_{miss}$ represents the FoV mismatch frequency between the client's requests and what has been transmitted.

At first, we present the result for how frequently the requested FoV is unavailable (FoV mismatch) in figure 7(a) within $100s$. Obviously, transmitting the full sphere(DF) never gets any mismatch. Compared with other settings of FoV, AF achieve an excellent result with fewer mismatches than the other strategies, and it is more stable with the change of link rate. The performance of AF is affected by the change range of the network delay and the network rate, thus the mismatch frequency is relatively low when the link rate is only $500bps$. In addition, a prediction method could also be applied in the algorithm of AF for choosing of the FoV more precisely.

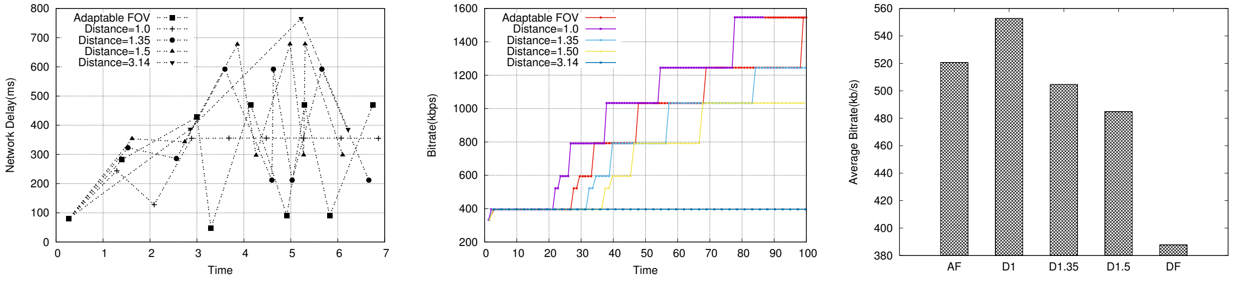
Figure 7(b) shows the result of QoE score of the five different scheme with normalization. When link rate is $1000kbps$, AF received the highest score compared with others, yet the scores of D1.35 and D1.5 was about the same, both around 5.9. DF shows a relatively good performance in QoE score since it has low mismatch frequency. Furthermore, when the link rate equals to $500kbps$, D1 has the lowest QoE score since the mismatch happens quite frequently. In this situation, there is a larger gap between AF and the other four cases, as AF jointly balances the bitrate and the FoV size. The results emphasize that the proposed algorithm has the ability to measure the impact of the network conditions on FoV adaptation to enhance the QoE of the users.

6 CONCLUSIONS

We have proposed a joint FoV and rate adaption mechanism for 360 degree video streaming that takes into account the responsiveness of the network. Indeed, if the content is delivered promptly from the server, the FoV can be reduced as the current view on the display will be more predictive of the view that is being requested.

We have observed that the users' movements are not rapidly changing in most cases. This offers the possibility for adapting the FoV according to the measured network delay in a short time interval. With the estimation of network latency, we calculate the segment length and the associated FoV size which could be delivered within the network delay, which is to say, so as to avoid the waiting time and mismatched view in the client side. Then we adjust the sending rate according to the network bandwidth, so as to achieve a high score of QoE.

In our simulations, such mechanism improves the sending bit rate significantly, an improvement of up to 1.34 × better resolution when compared with the full spherical 360 degree video stream. Moreover, when rate adaption is used, it could achieve best QoE in different network conditions by considering FoV and delay together.

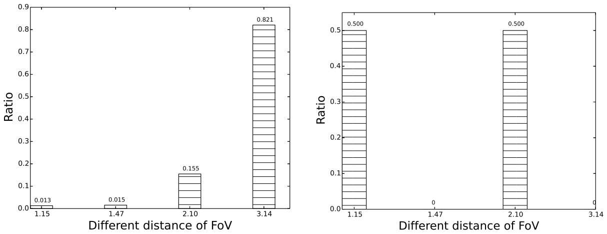


(a) Network delay variations with different settings of FoV

(b) Bitrate variation with different settings of FoV

(c) Average Bitrate

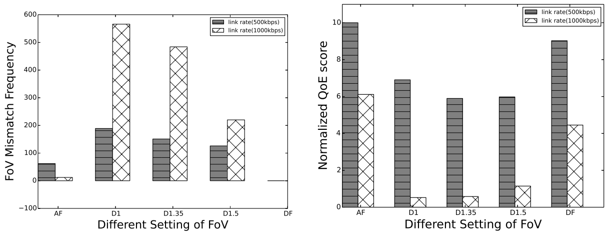
Figure 5: Performance in network delay and bitrate



(a) Ratio of different of FoV (link-rate 500kbps)

(b) Ratio of different of FoV (link-rate 1000kbps)

Figure 6: The ratio of different of FoV



(a) Frequency of FoV mismatch

(b) Normalized QoE score

Figure 7: Frequency of FoV mismatch and QoE score

Further work includes optimizing the function of getting FoV size with state of art prediction models, including directional models. Implementing the system in an actual video client is left as future work as well.

7 ACKNOWLEDGEMENTS

We are grateful to the anonymous reviewers for their insights. We also thank Jinlei Jiang and Guangwen Yang for their valuable support of this work. In addition, Dongbiao He is supported by the Tsinghua Scholarship for Overseas Graduate Studies. This research was supported in part by the National Natural Science Foundation of China (Grant No. 61572280).

REFERENCES

[1] Shahryar Afzal, Jiasi Chen, and K. K. Ramakrishnan. 2017. Characterization of 360-degree Videos. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network (VR/AR Network '17)*. ACM.

[2] NGMN Alliance. 2015. NGMN 5G White Paper. https://www.ngmn.org/uploads/media/NGMN_5G_White_Paper_V1_0.pdf. (Feb. 2015).

[3] Stephen Cass and Charles Choi. 2015. Google Glass, HoloLens, and the Real Future of Augmented Reality. In *IEEE Spectrum*.

[4] Jacob Chakareski. 2017. VR/AR Immersive Communication: Caching, Edge Computing, and Transmission Trade-Offs. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network*. ACM, 36–41.

[5] Jacob Chakareski, Ridvan Aksu, Xavier Corbillon, Gwendal Simon, and Viswanathan Swaminathan. 2018. Viewport-Driven Rate-Distortion Optimized 360° Video Streaming. *arXiv preprint arXiv:1803.08177* (2018).

[6] Xavier Corbillon, Francesca De Simone, and Gwendal Simon. 2017. 360-degree video head movement dataset. In *Proceedings of the 8th ACM on Multimedia Systems Conference*. ACM, 199–204.

[7] Xavier Corbillon, Alisa Devlic, Gwendal Simon, and Jacob Chakareski. 2017. Optimal Set of 360-Degree Videos for Viewport-Adaptive Streaming. *in Proc. of ACM Multimedia (MM)* (2017).

[8] Xavier Corbillon, Gwendal Simon, Alisa Devlic, and Jacob Chakareski. 2017. Viewport-adaptive navigable 360-degree video delivery. In *Communications (ICC), 2017 IEEE International Conference on*. IEEE, 1–7.

[9] Fanyi Duanmu, Eymen Kurdoglu, S Amir Hosseini, Yong Liu, and Yao Wang. 2017. Prioritized buffer control in two-tier 360 video streaming. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network*. ACM, 13–18.

[10] F. Duanmu, E. Kurdoglu, Y. Liu, and Y. Wang. 2017. View direction and bandwidth adaptive 360 degree video streaming using a two-tier system. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*.

[11] Tarek El-Ganainy. 2017. Spatiotemporal Rate Adaptive Tiled Scheme for 360 Sports Events. *CoRR* abs/1705.04911 (2017). http://arxiv.org/abs/1705.04911

[12] Mohammad Hosseini and Viswanathan Swaminathan. 2016. Adaptive 360 VR Video Streaming based on MPEG-DASH SRD. In *IEEE International Symposium on Multimedia (ISM'16)*.

[13] Xing Liu, Qingyang Xiao, Vijay Gopalakrishnan, Bo Han, Feng Qian, and Matteo Varvello. 2017. 360 Innovations for Panoramic Video Streaming. In *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*. ACM, 50–56.

[14] Simone Mangiante, Guenter Klas, Amit Navon, Zhuang GuanHua, Ju Ran, and Marco Dias Silva. 2017. Vr is on the edge: How to deliver 360 videos in mobile networks. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network*. ACM, 30–35.

[15] Kazuhisa Matsuzono, Hitoshi Asaeda, and Thierry Turletti. 2017. Low latency low loss streaming using in-network coding and caching. In *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*. IEEE, 1–9.

[16] Djuro Mirkovic, Grenville Armitage, and Philip Branch. 2018. A survey of Round Trip Time Prediction Systems. *IEEE Communications Surveys & Tutorials* (2018).

[17] Feng Qian, Bo Han, Lusheng Ji, and Vijay Gopalakrishnan. 2016. Optimizing 360 Video Delivery Over Cellular Networks. In *ACM MobiCom All Things Cellular Workshop*.

[18] Cedric Westphal. 2017. Challenges in Networking to Support Augmented Reality and Virtual Reality. In *IEEE ICNC*.

[19] Liyang Zhang, Syed Obaid Amin, and Cedric Westphal. 2017. VR video conferencing over named data networks. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network*. ACM, 7–12.