Efficient, QoE aware delivery of 360^0 videos on VR headsets over mobile links

Apoorv Saxena
Saxena.Apoorv@Ugent.be
SMACS Research Group, TELIN
Ghent University
Gent, Belgium

Shishir Subramanyam
S.Subramanyam@cwi.nl
Distributed and Interactive Systems, CWI
Amsterdam, Netherlands

Pablo Cesar
P.S.Cesar@cwi.nl
Distributed and Interactive Systems, CWI
Amsterdam, Netherlands

Rob van der Mei
R.D.van.der.Mei@cwi.nl
Stochastics, CWI
Amsterdam, Netherlands

Berg, J.L. (Hans) van den
j.l.vandenberg@tno.nl
Stochastics, CWI
Amsterdam, Netherlands

ABSTRACT

Virtual reality headsets have been in a great demand in the past years due to the immersive experience it can now provide to the users. An immersive experience of high quality, however, requires a high bandwidth connection. The latest generation of these headsets support wireless connectivity and rely on a battery to power the headset. With advancements in 5G technology, soon these headsets would be able to connect to the network directly. To provide a similar immersive and uninterrupted experience to the users, the headsets would need to take into account the intermittent behavior of the wireless channel.

In this paper we model the streaming of 360^0 videos on a VR headset using a wireless channel. We model both the intermittent behavior of the wireless channel as well as the prediction of the head position of the user. The MDP based streaming algorithm we present, guarantees a high Quality of Experience (QoE) for the headset users while ensuring that the resource usage is minimised.

CCS CONCEPTS

• **Human-centered computing** → **User models**; **Virtual reality**.

KEYWORDS

VR, 5G, FoV, Buffering, Markov Decision Processes, QoE

ACM Reference Format:
Apoorv Saxena, Shishir Subramanyam, Pablo Cesar, Rob van der Mei, and Berg, J.L. (Hans) van den. 2020. Efficient, QoE aware delivery of 360^0 videos on VR headsets over mobile links. In *13th EAI International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS '20), May 18–20, 2020, Tsukuba, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3388831.3388834

VALUETOOLS '20, May 18–20, 2020, Tsukuba, Japan

ACM ISBN 978-1-4503-7646-4/20/05…$15.00
https://doi.org/10.1145/3388831.3388834

1 INTRODUCTION

VR headsets have gained tremendous popularity in the last few years and this interest is expected to increase even further in the near future. The potential applications of this technology range from education [3], museum tours [15], medical training industry [18] to gaming industry [4]. We are primarily interested in the use of VR headsets to stream 360^0 videos commonly used due to the powerful immersive experience it provides to the users [5]. 5G networks would be able to provide high network bandwidth which is required to operate these headsets. The primary focus of this paper is therefore to analyze and improve streaming of 360^0 videos on VR headsets particularly when the network conditions are unreliable.

A VR headset requires a high reliability and robustness against the fluctuations of the channel quality since the latency requirements are between 6 - 15 ms. A latency higher than 15 ms not only degrades the viewing quality, it also leads to issues like motion sickness [9]. These bandwidth and latency requirements are already stretching our abilities to provide network services. Moreover, due to the increasing popularity, big companies like Facebook and Youtube have invested heavily in VR streaming. Therefore, we expect to see even more increase in the usage by regular users as VR headsets

Figure 1: Oculus rift

integrate into everyday use. This increase in the number of users will put even more stress on the network.

Current capabilities of wireless communication under 4G do not support enough bandwidth to be able to handle wireless streaming on VR headsets. Therefore, the streaming services are dependent on a wired Ethernet connection. The same cable is used to supply the power to the headset (see Figure 1. However, with the growth of 5G wireless standards, it will be possible to provide the services without a wired connection. Wireless transmission is desired for two main reasons:

- Being connected to a wire results in a degraded immersive experience as it limits the free movement.
- The wires can also be dangerous as one can easily trip on them as their vision is completely occupied by the headsets.

Streaming completely wireless comes with a couple of challenges. First, wireless communication is less reliable. That is, 5G has more intermittent behavior due to shadowing etc. Secondly, in the absence of any cable, the headset would have to be powered by a battery. Therefore, the consumption of battery power, which is linked to the number of times wireless antenna is used, has to be minimal. To tackle these challenges, we need to maintain a buffer of the correct tiles by pre-fetching when the wireless channel quality is good. When the channel quality is bad, this buffer can mitigate the impact and maintain the high standards of QoE of the users.

In normal videos, when video needs to be buffered, the complete view of the video is downloaded in the best possible resolution permitted by the available bandwidth. Therefore, the buffer length is simply the length of video in the buffer which can be played back. However, in 360^0 videos only the portion of the video that is in the view of the user is transmitted. This portion of the video is called the Field of View (FoV) and accounts for less than 15% of the total video (see figure 2). Such transmission reduces the bandwidth requirements which makes the high quality FoV transmission possible.

The buffering mechanism for 360^0 videos is therefore much more challenging than for a simple video. Not only do we need to account for the network interruptions, we also need

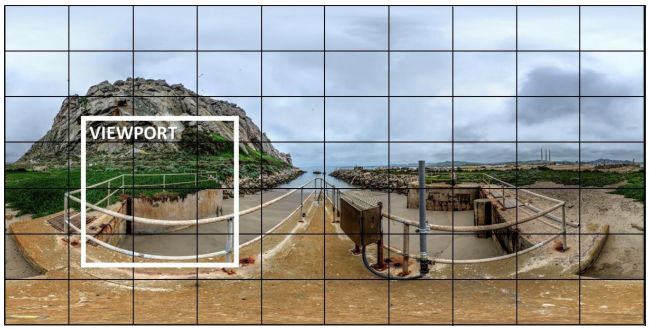

Figure 2: The FoV of a user, see [19] for more details.

to take into account the user head movements to build a useful buffer. Additionally, if the user head moves in an unanticipated direction, the buffer needs to be rebuilt. Since the device is supported by a battery and a wireless network connection, we need to ensure that both these resources are used reasonably.

Providing such an immersive experience comes with a lot of challenges. Since the video outside the FOV is transmitted in the lowest quality, there is a risk that if the user moves outside the anticipated FoV, a low quality fallback video is displayed. Many papers have tried to optimize the different components of streaming 360^0 videos. [17] aims to minimize the average transmission rate by addressing the trade-off between creating projections at the VR headsets versus doing it at the edge node. [22] look at a similar trade off with delay constraints by utilizing computation and buffering resources at the VR device. [20] optimize the buffer use by storing video in multiple bit rates.

A few papers have analysed the transmission of 360^0 videos on the headset. [8] use linear regression to predict and download the FoV of the user up to two seconds and download tiles of this predicted FoV. [12] implement a methodology where the tiles in the FoV of user and streamed at the highest possible quality and a lowest for tiles outside this view with buffer length limited to two seconds. [10] maintain an additional low quality buffer of the whole 360^0 video to avoid scenario in which nothing is shown to the user. [16] investigate a multi-tier approach of streaming on VR headsets, where the impact of bad network conditions are mitigated by falling back on a wifi connection. All the studies use simulations to analyze the performance and comparison with other strategies. While simulation studies may have some advantages, performing them is a very time consuming exercise. On the other hand, models are able to provide a precise table of download decisions the headset must take encompassing all the system scenarios. Moreover, models also guarantee that

the long run cost of performing the task would be minimal.

In this paper, we use the patterns of head movement of the user and the information about the interesting parts of the video to build the buffer which has FoV in the highest quality. We model this as a Markov Decision Process (MDP) which helps us decide which parts of the video should be buffered. For example, if enough high quality tiles are not available for the upcoming video segment, the headsets attempts to use as much bandwidth that it is permitted to use in the given channel quality. However, if the upcoming segments of FoV are available in high quality, tiles are downloaded only if the channel quality is very good, as downloading in such conditions uses fewer resources. More precisely, downloading decisions are taken to build a buffer in a way such that the trade-off between high QoE and resource usage is optimized. We show how the buffering mechanism translates to MDP parameters. The MDP approach leads to a downloading policy that ensures a proper trade-off between QoE and resource usage.

Our contribution is therefore:

- We show that a higher quality of FoV can be achieved using our approach at a lower expected cost.
- The mechanism allows the flexibility to model different video types and users to accurately measure and optimize the cost and QoE of each user.
- Unlike simulations, using the MDP model, the buffer decisions can be computed in a few minutes.

We start with an overview of streaming of videos in Section 2 where we provide the required details of streaming process on VR headsets. We discuss the modelling assumptions in Section 3. This is followed by an overview of MDP formulation of the problem in Section 4 where we show how we model the primary features of VR streaming of 360^0 videos. In the Sections 5 - 7 we evaluate the performance of this buffering approach by comparing it with other common approaches.

2 VIDEO STREAMING ON VR

360^0 videos are divided into small tiles which can be downloaded separately and then stitched together after the transmission. Since only the FoV tiles are primarily required, those tiles are rendered in the highest resolution possible. The tiles outside the FoV are transmitted at the lowest resolution as a fall back mechanism. Therefore, any buffering mechanism needs to predict and account for the FoV of the user to build an appropriate buffer. However, if the user makes an unanticipated movement, this buffer is rendered useless and needs to be rebuilt.

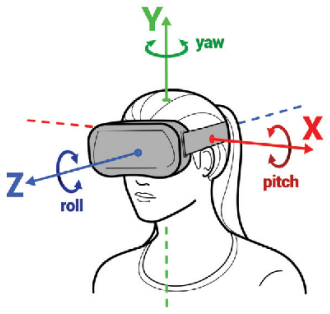

Figure 3: 360^0 video viewing direction from [13]

Video is transmitted in short segments of a fraction of second. That is, if a normal video is buffered, the transmission is done to add short segments of the video to build the buffer. Similarly, in 360^0 videos, the video is segmented in short segments in time. Therefore, when a tile is downloaded a short segment of that portion of video is downloaded. In this paper we assume that the segments are half a second long.

Buffering mechanism is case of 360^0 videos is challenging, because we need to predict the FoV of the user. If incorrect tiles are downloaded, it leads to higher costs without any improvement to QoE. When buffering we have two types of information available about the user interest namely short predictions and long predictions.

Short term prediction

At any point in time, we have access to the head position of the user and the angular momentum of their head in three independent directions (see figure 3). Using this information, it is possible to predict the head position in the next one or two seconds with very high accuracy. [14] show that accurate predictions are possible upto half or one seconds with accuracy of more than 92%. More methods exist for such predictions. For example, [11] describe a gravitational predictor and use of AI-techniques to predict the interesting part of the video. [7] use supervised learning to predict the eye position.

In our paper, we assume these predictions are represented as three-dimensional distribution function and are known beforehand. That is, $P(\theta_r, \theta_p, \theta_y)$ is the probability that the head moves by θ_r, θ_p and θ_y in the roll, pitch and yaw (see figure 3). Note that these predictions are user dependent and are only accurate for one or two seconds. Therefore, we can only use them to build the buffer of the first few segments.

Long term prediction

If we observe different users over same video segments, there is a lot of overlap in their viewing patterns. That is, different

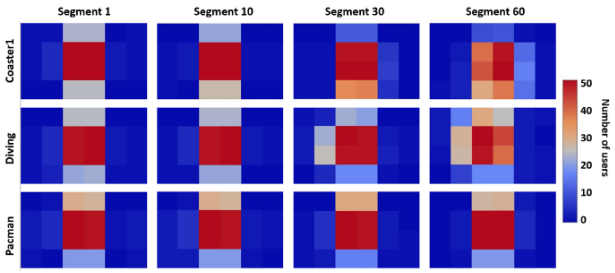

Figure 4: Heatmap of user's FoV pattern from [8]

users find similar areas of the videos interesting and their FoV is in that general direction. [21] provide a data-set of head movements of viewers watching 360^0 videos. [2] analyze the traces of viewing data of more than 150 users and conclude that the viewer's focus follows a similar patter over all users using the data shared in [1]. [6] use traces of head movement of users from 19 VR videos to predict the future FoV position of the user using deep learning.

Note: For each segment, the tiles have a given probability of being in the FoV. These probabilities define the sequence in which these tiles are downloaded. Therefore, to know the status of a future segment in the buffer, we only need to know the number of tiles downloaded in the buffer for that segment. When a user moves the head, this probability distribution changes which makes only a fraction of tiles in the buffer useful.

3 MODELLING ASSUMPTIONS

In the following sections, we will model the buffering process as a discrete-time Markov decision process. That is, time is divided into slots of a fraction of a second and the decisions to buffer are taking at the beginning of a slot. We assume that the channel quality is a Markov process, described by probability transition matrix P_{CQ}. It is assumed that the channel quality remains constant during a single video segment. The aim is to perform buffer operations at the beginning of these slots when the channel quality is good, such that the FoV is available for future segments in a high quality. The algorithm optimizes a weighted average of QoE and resource usage.

We will make the following assumptions:

- The whole 360^0 video is always available in the lowest quality. We do not consider this transmission as a part of our problem as it doesn't require much bandwidth.
- Tiles of the best quality are transmitted to improve the quality of the FoV. Our focus is only on the transmission of these high quality tiles.

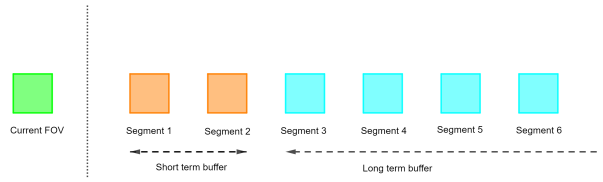

Figure 5: Buffer divided into short term and long term buffer

- The user behaviour is known from the beginning, i.e., the distribution of the navigation pattern $P(\theta_r, \theta_p, \theta_y)$ is known.

We divide the buffer into two parts (see figure 5)

- Short-term buffer which is limited to the first two segments
- Long-term buffer which comprises of all the segments after the short term buffer.

The head movements only influence the first two segments and are always given the higher priority. Moreover, we enforce that for long-term buffer segments, no more than M tiles are downloaded for each segment. This is done to ensure that too many tiles are not downloaded using long term predictions as they are not user specific.

4 MDP FORMULATION FOR BUFFERING MECHANISM

A Markov decision process has four major components: the state space, the probability transition matrix, the action space and the reward function. To capture the buffering mechanism, we need to define the condition of the buffer and channel in the form of a state. This state should include the information of the number of tiles in the buffer, the time left in the current segment being played as well the condition of the channel quality. Let us denote the state space as $\mathbb{S}$, the current slot number is denoted by k and that the system is in state s_1 at the starting of this slot.

At the beginning of each slot, a buffer decision can be made out of all possible actions denoted by action set $\mathbb{A}$. The validity of these actions depends on the state of the system. Let us assume that the action chosen is denoted by a at the beginning of slot k.

The system transitions between states as new tiles are downloaded and channel quality changes. This transition is also dependent on the action a chosen at the beginning of the slot. The matrix $P(s_2|s_1, a)$ defines the probability that the system enters state s_2, *given* that it was in state s_1 and the action a was performed.

The system is rewarded for being in any given state, where the states which ensure that the QoE is good have a higher reward. This influences the decision of the buffering process to move the system closer to the high reward states. This reward function is defined as $R(a, s_1)$, that is, $R(a, s_1)$ is the amount of reward received if an action a is chosen if the system is in state s_1. Precisely, it is a weighted sum of the quality of the predicted FoV defined by s_1 and the direct cost of performing the action a.

As the system transitions between these states and takes actions, it collects rewards. Therefore, using for any given state, we can define the expected utility for each state of a given policy π as:

$$V^{\pi}(s) = R(\pi(s), s) + \sum_{s'} \gamma P_{\pi}(s', s) V^{\pi}(s')$$

Therefore, using the framework of the Markov decision processes, we want to arrive at the set of actions which would lead to the maximum expected total reward of the system. This is because these set of actions will influence the system to move to states with higher reward which are representative of high QoE and low expected cost. Such a set of action space is called the *optimal policy*. We will use π^* to define such an optimal policy where

$$\pi^*(s) = a^* \tag{1}$$

for each state s it gives you the optimal action a^*.

State Space. The state space is defined by

$$\begin{aligned}\mathbb{S} = \{&(n_1, n_2, L, N, R, CQ) \\ &0 \le n_1 \le T, 0 \le n_2 \le T, 0 \le L \le L_{max}, \\ &0 \le N \le M, 1 \le R \le Q, 0 \le CQ \le CQ_{max}\},\end{aligned} \tag{2}$$

where n_1 is the number of tiles in the first buffer segment, n_2 is the number of tiles in the second buffer segment, L is the number of long-term segments which have already been downloaded and N is the number of tiles in the current long term segment which needs more tiles. R is the number of slots left in the current video segment being played at the headset and CQ is the channel quality during the slot. At the end of the video segment in the headset, the tiles in the first buffer segment are used to build the FoV of the user.

Action space. The action space defines all the buffer actions that can be taken when the system is in a given state. We define actions using a tuple (bs, n), where bs is the buffer segment in which the tiles are downloaded and n is the number of tiles downloaded. For the short term buffer, $bs = 1$ or 2, while for long term buffer, $bs = 3$. That is

$$\mathbb{A} = \{(0, 0)\} \cup \{(bs, n) : bs = 1, \ldots 3, n = 1, \ldots T\} \tag{3}$$

All the actions may not be valid in a given state. For example, if the channel quality $CQ = 0$, no tiles can be downloaded. Therefore, any action $(bs, 0)$ with $bs > 0$ is not valid.

Notation List	
s_1, s_2, s	used to denote states
$\mathbb{S}$	the state space
a	used to denote actions
$\mathbb{A}$	the action space
n_1, n_2	used to denote the number of tiles in segment one and two
T	maximum number of tiles that can be downloaded in a segment
CQ	the channel quality
CQ_{max}	highest channel quality state possible
P_{CQ}	probability transition matrix of channel quality
N	number of tiles in the current long-term buffer segment
M	maximum number of tiles that can be downloaded in any long-term segment
R	remaining number of slots in the current video segment
Q	length of each video segment
L	number of long-term segments downloaded in buffer
L_{max}	maximum number of long-term segments that can be downloaded in buffer
(bs, n)	action which means n tiles are download in segment bs
$R(a, s)$	reward gained by taking an action a in state s
$P(s_2 \mid a, s_1)$	probability that their is a state change to s_2 given that action a is taken while system is in state s_1

Probability transition matrix. The system transitions between different states based on the following factors

- the action chosen in the previous slot: for instance, if more tiles are downloaded in the second segment, system would transition to a state with higher n_2.
- user head movements: for instance, if the user moves their head a few degrees more than anticipated, some of the tiles in the buffer would become useless. That is, out of n_1 and n_2 tiles in the buffer, only a fraction of them would be useful. Therefore, the system would move to a state with smaller n_1 and n_2 depending on the magnitude of movement.
- channel quality: the channel quality, alone, is assumed to follow a Markov process. This transition is determined by the probability transition matrix P_{CQ}.

- remaining time in the current segment of the video: when the current segment of headset is over, the first segment of the buffer is used to create the FoV of the user. Which means that in the next slot, n_1 is replaced by n_2 and n_2 is replaced by M or N depending on the state of long term buffer segments.

Reward function. In a given state s, if action a is chosen, it would lead to an award of $R(a, s)$. This reward is a weighted sum of two quantities:

- *Cost of downloading*: downloading tiles in a channel state incurs a cost which increases as the channel quality degrades. Therefore, algorithm is penalized higher if tiles are downloaded in a worse channel quality.
- *FoV quality*: a state defines the quality of the predicted FoVs. Therefore, the algorithm is penalised if the buffer of the predicted FoVs is not good. Moreover, the first buffer segment is prioritized over the second buffer segment which has a higher priority over long term segments. Therefore, the penalty for FoV unavailability decreases with higher segment number.

By choosing such penalties/rewards we are able to prioritise the tiles in the short-term segments. Moreover, they are downloaded when the channel quality is good to avoid possible future higher costs.

Policy iteration. Policy iteration consists of two sequential phases, policy evaluation and policy improvement. In policy evaluation, we start with a random policy π and evaluate the policy using the Belmann equation.

After the end of policy evaluation, we check if a better action is available for each state. The final improved policy is the optimal policy π^* which optimises a weighted sum of costs and QoE.

Result: A stable policy
$\pi(s) \in \mathbb{A}$ and $V(s) \in \mathbb{U}[0, 100]$ arbitrarily for all $s \in \mathbb{S}$
while $\Delta > \theta$ **do**
  $\Delta \leftarrow 0$
  foreach $s \in \mathbb{S}$ **do**
    $t \leftarrow V(s)$
    $V(s) \leftarrow \sum_{s'} p(s'|s, \pi(s))\Big(r(s, \pi(s), s') + \gamma V(s')\Big)$
    $\Delta \leftarrow max(\Delta, |t - V(s)|)$
  end
end

Algorithm 1: Policy evaluation

5 ALGORITHM PARAMETERS

To view the parameters and the exact code, have a look at the github repository[1].

Result: Optimal policy
policy-stable ← False
while *policy-stable* ≠ *True* **do**
  policy-stable ← True
  foreach $s \in \mathbb{S}$ **do**
    $t \leftarrow \pi(s)$
    $\pi(s) \leftarrow$
    $\arg\max_a \sum_{s'} p(s'|s, a)\Big(r(s, a, s') + \gamma V(s')\Big)$
    if $t \neq \pi(s)$ **then** policy-stable ← False ;
  end
end

Algorithm 2: Policy improvement

FoV tile distribution. Downloading more tiles improves the quality of FoV, however, the quality improvement is much higher when there are few tiles in the buffer. Therefore, we assume that the cumulative probability distribution of the utility of tiles is like cumulative exponential distribution.

Head movements. In compliance with the studies about tracking the head-movements of users, we assume that with about 0.95 probability, our predictions of FoV are accurate. With probability about 0.05, the FoV is uniformly distributed over the whole range of possible angular movements.

Channel quality. The authors of [16] use 5G traces to measure the performance of a multi-tier approach of streaming on VR headsets. These traces suggest that using 3 states for channel quality would be enough to measure the performance of the buffering policy. The model can however support any general channel quality matrix.

Long term predictions. Our model requires only the parameter M to be defined for long term predictions. A high values of M may result in wastage of resources, while using a very small value defeats the purpose of using long term predictions. We use $M = 80\%$ of the maximum number of tiles we need to complete a FoV.

6 PERFORMANCE MEASURES

We compute two main performance measures:

- FoV quality: as high quality tiles are downloaded, the quality of FoV improves. This measure measures the quality of the FoV by looking at the buffer at the end of the video segment begin played on the headset.

$$\text{FoV quality} = \frac{1}{K}\sum_{k=1}^{K} F(s_k.n1 \big| \text{segment ends at slot } k) \quad (4)$$

where s_k is the state of the system at the end of slot k and $F()$ is the FoV tile distribution defined in section 5 and K is the length of the simulation.

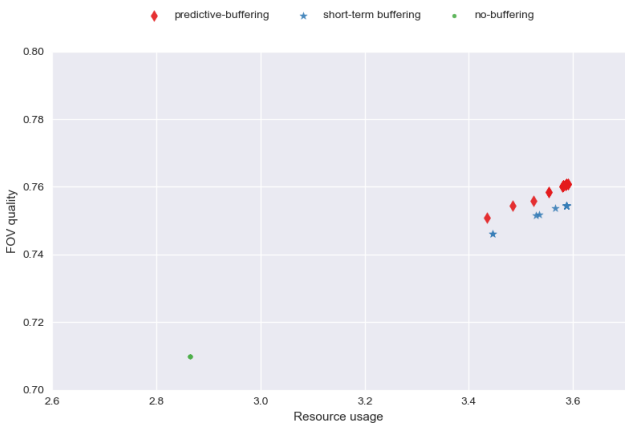

Figure 6: Scatter plot of QoE vs cost

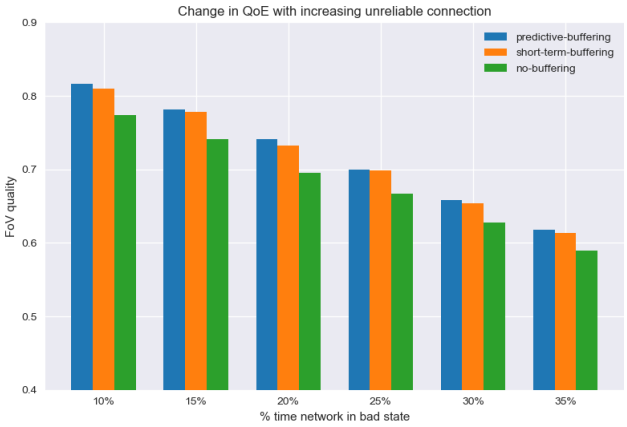

Figure 7: Change in QoE with channel unreliability

- Cost of streaming: Downloading tiles in different channel conditions comes at a different cost. The idea is to improve the FoV quality at the least additional cost. This measure captures the cost of streaming using any policy.

$$\text{Cost} = \frac{1}{K}\sum_{k=1}^{K} \text{Number of tiles}(\pi(s_k)) * Cost(s_k.CQ) \quad (5)$$

where π is the streaming policy which gives us the chosen actions. In case of predictive buffering, we use the optimal policy obtained by policy iteration defined in section 4. s_k the state of the system at the end of slot k and $Cost()$ is the cost of downloading a single tile in a given channel quality.

To numerically analyze the performance, we compare three streaming policies[1]

- *No-buffering*: In this methodology, the headset does not plan any buffer segments. While the current segment is played out, the tiles are downloaded for just the upcoming segment. This strategy is the greedy approach which does not consider the variation in the channel quality and the cost of downloading in a bad channel state.
- *Predictive-buffering*: this is the approach which uses the optimal MDP policy that optimizes the weighted sum of resource usage and QoE. Both short-term and long-term segments are maintained in the buffer with short-term segments receiving a higher priority over long-term segments.
- *Short-term buffering*: In this methodology, only a short-term buffer is maintained. That is, only the head movement statistics are utilized to download the tiles. This approach is basically previous approach (predictive buffering) with $L_{max} = 0$ as no tiles are downloaded for long-term segments.

[1]The code is available at https://github.com/saxe405/buffer

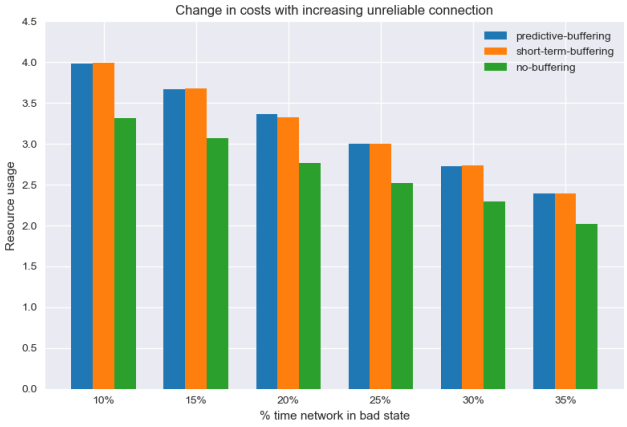

Figure 8: Change in cost with channel unreliability

7 NUMERICAL RESULTS

In this section, we briefly discuss the performance of the three streaming policies described in the previous section.

- In Figure 6, we show scatter plot of resource usage and quality of FOV. Since there is no caching mechanism in place for 'no-buffering' scenario, we have a single point. However, for 'short-term buffering' and 'predictive-buffering', we can have different weights for cost of downloading and FOV quality. Therefore, we have multiple points for these scenarios, each defined by a different relative weight of cost of downloading and FOV quality. There is an improvement of about 10% in quality just by using the short term predictions. Since more high quality tiles are downloaded

to maintain a buffer, this improvement is possible at an additional cost. Further, by using both long term and short term predictions, the quality of FoV is further improved without a noticeable increase in the average cost.

- In Figure 7 and 8 we illustrate the change in quality of FOV and resource usage as the network becomes increasingly unreliable. We can observe that the improvement in the FOV quality by using long term predictions happens at no additional long run cost. This is because the long term buffer tiles are downloaded only when the cost of networking is very low and the short term buffer tiles have already been downloaded.

8 FUTURE WORK

We are performing experiments to measure the head movements of the users. This data-set will be used to classify users into different categories. Further, the transmission quality can also depend on the type of the video being viewed. We will work on including different user types and video types in our model.

REFERENCES

[1] Y. Bao, H. Wu, T. Zhang, A. A. Ramli, and X. Liu. 2016. Shooting a moving target: Motion-prediction-based transmission for 360-degree videos. In *2016 IEEE International Conference on Big Data (Big Data)*. 1161–1170. https://doi.org/10.1109/BigData.2016.7840720

[2] Y. Bao, T. Zhang, A. Pande, H. Wu, and X. Liu. 2017. Motion-Prediction-Based Multicast for 360-Degree Video Transmissions. In *2017 14th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 1–9. https://doi.org/10.1109/SAHCN.2017.7964928

[3] Abbie Brown and Tim Green. 2016. Virtual Reality: Low-Cost Tools and Resources for the Classroom. *TechTrends* 60, 5 (01 Sep 2016), 517–519. https://doi.org/10.1007/s11528-016-0102-z

[4] J. C. P. Chan, H. Leung, J. K. T. Tang, and T. Komura. 2011. A Virtual Reality Dance Training System Using Motion Capture Technology. *IEEE Transactions on Learning Technologies* 4, 2 (April 2011), 187–195. https://doi.org/10.1109/TLT.2010.27

[5] Ahmed Elmezeny, Nina Edenhofer, and Jeffrey Wimmer. 2018. Immersive Storytelling in 360-Degree Videos: An Analysis of Interplay Between Narrative and Technical Immersion. *Journal For Virtual Worlds Research* 11, 1 (2018). https://doi.org/10.4101/jvwr.v11i1.7298

[6] Xueshi Hou, Sujit Dey, Jianzhong Zhang, and Madhukar Budagavi. 2018. Predictive View Generation to Enable Mobile 360-degree and VR Experiences. In *Proceedings of the 2018 Morning Workshop on Virtual Reality and Augmented Reality Network (VR/AR Network '18)*. ACM, New York, NY, USA, 20–26. https://doi.org/10.1145/3229625.3229629

[7] T. Judd, K. Ehinger, F. Durand, and A. Torralba. 2009. Learning to predict where humans look. In *2009 IEEE 12th International Conference on Computer Vision*. 2106–2113. https://doi.org/10.1109/ICCV.2009.5459462

[8] Anahita Mahzari, Afshin Taghavi Nasrabadi, Aliehsan Samiei, and Ravi Prakash. 2018. FoV-Aware Edge Caching for Adaptive 360 Video Streaming. In *Proceedings of the 26th ACM International Conference on Multimedia (MM '18)*. ACM, New York, NY, USA, 173–181. https://doi.org/10.1145/3240508.3240680

[9] Simone Mangiante, Guenter Klas, Amit Navon, Zhuang GuanHua, Ju Ran, and Marco Dias Silva. 2017. VR is on the Edge: How to Deliver 360&Deg; Videos in Mobile Networks. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network (VR/AR Network '17)*. ACM, New York, NY, USA, 30–35. https://doi.org/10.1145/3097895.3097901

[10] Afshin Taghavi Nasrabadi, Anahita Mahzari, Joseph D. Beshay, and Ravi Prakash. 2017. Adaptive 360-Degree Video Streaming Using Scalable Video Coding. In *Proceedings of the 25th ACM International Conference on Multimedia (MM '17)*. ACM, New York, NY, USA, 1689–1697. https://doi.org/10.1145/3123266.3123414

[11] Oculus. 2018. https://code.fb.com/virtual-reality/enhancing-high-resolution-360-streaming-with-view-prediction/.

[12] Stefano Petrangeli, Viswanathan Swaminathan, Mohammad Hosseini, and Filip De Turck. 2017. An HTTP/2-Based Adaptive Streaming Framework for 360 Virtual Reality Videos. In *Proceedings of the 25th ACM International Conference on Multimedia (MM '17)*. ACM, New York, NY, USA, 306–314. https://doi.org/10.1145/3123266.3123453

[13] Feng Qian, Lusheng Ji, Bo Han, and Vijay Gopalakrishnan. 2016. Optimizing 360 video delivery over cellular networks. In *ATC@MobiCom*.

[14] Feng Qian, Lusheng Ji, Bo Han, and Vijay Gopalakrishnan. 2016. Optimizing 360 Video Delivery over Cellular Networks. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges (ATC '16)*. ACM, New York, NY, USA, 1–6. https://doi.org/10.1145/2980055.2980056

[15] Guy Schofield, Gareth Beale, Nicole Beale, Martin Fell, Dawn Hadley, Jonathan Hook, Damian Murphy, Julian Richards, and Lewis Thresh. 2018. Viking VR: Designing a Virtual Reality Experience for a Museum. In *Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18)*. ACM, New York, NY, USA, 805–815. https://doi.org/10.1145/3196709.3196714

[16] Liyang Sun, Fanyi Duanmu, Yong Liu, Yao Wang, Yinghua Ye, Hang Shi, and David Dai. 2018. Multi-path Multi-tier 360-degree Video Streaming in 5G Networks. In *Proceedings of the 9th ACM Multimedia Systems Conference (MMSys '18)*. ACM, New York, NY, USA, 162–173. https://doi.org/10.1145/3204949.3204978

[17] Yaping Sun, Zhiyong Chen, Meixia Tao, and Hui Liu. 2018. Communication, Computing and Caching for Mobile VR Delivery: Modeling and Trade-Off. In *2018 IEEE International Conference on Communications, ICC 2018, Kansas City, MO, USA, May 20-24, 2018*. 1–6. https://doi.org/10.1109/ICC.2018.8422519

[18] Oren M. Tepper, Hayeem L. Rudy, Aaron Lefkowitz, Katie A. Weimer, Shelby M. Marks, Carrie S. Stern, and Evan S. Garfein. 2017. Mixed Reality with HoloLens: Where Virtual Reality Meets Augmented Reality in the Operating Room. *Plastic and Reconstructive Surgery* 140, 5 (1 11 2017), 1066–1070. https://doi.org/10.1097/PRS.0000000000003802

[19] Tiledmedia. 2019. https://www.tiledmedia.com/index.php/technology/.

[20] F. Wang, Z. Fei, J. Zheng, and J. Wang. 2018. QoE-Aware Mobile VR HAS Cache Management With Coding Helper. *IEEE Access* 6 (2018), 44556–44569. https://doi.org/10.1109/ACCESS.2018.2864667

[21] Chenglei Wu, Zhihao Tan, Zhi Wang, and Shiqiang Yang. 2017. A Dataset for Exploring User Behaviors in VR Spherical Video Streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. ACM, New York, NY, USA, 193–198. https://doi.org/10.1145/3083187.3083210

[22] X. Yang, Z. Chen, K. Li, Y. Sun, N. Liu, W. Xie, and Y. Zhao. 2018. Communication-Constrained Mobile Edge Computing Systems for Wireless Virtual Reality: Scheduling and Tradeoff. *IEEE Access* 6 (2018), 16665–16677. https://doi.org/10.1109/ACCESS.2018.2817288