Analyzing Viewport Prediction Under Different VR Interactions

Tan Xu
AT&T Labs Research
Bedminster, NJ
tanxu@research.att.com

Bo Han
AT&T Labs Research
Bedminster, NJ
bohan@research.att.com

Feng Qian
University of Minnesota, Twin Cities
Minneapolis, MN
fengqian@umn.edu

ABSTRACT

In this paper, we study the problem of predicting a user's viewport movement in a networked VR system (*i.e.,* predicting which direction the viewer will look at shortly). This critical knowledge will guide the VR system through making judicious content fetching decisions, leading to efficient network bandwidth utilization (*e.g.,* up to 35% on LTE networks as demonstrated by our previous work) and improved Quality of Experience (QoE). For this study, we collect viewport trajectory traces from 275 users who have watched popular 360° panoramic videos for a total duration of 156 hours. Leveraging our unique datasets, we compare viewport movement patterns of different interaction modes: wearing a head-mounted device, tilting a smartphone, and dragging the mouse on a PC. We then apply diverse machine learning algorithms – from simple regression to sophisticated deep learning that leverages crowd-sourced data – to analyze the performance of viewport prediction. We find that the deep learning approach is robust for all interaction modes and yields supreme performance, especially when the viewport is more challenging to predict, *e.g.,* for a longer prediction window, or with a more dynamic movement. Overall, our analysis provides key insights on how to intelligently perform viewport prediction in networked VR systems.

CCS CONCEPTS

• **Information systems** → **Multimedia streaming**; • **Human-centered computing** → **Virtual reality**; • **Computing methodologies** → **Machine learning**; • **Networks** → *Mobile networks.*

KEYWORDS

360-degree video; VR interactions; viewport prediction; adaptive video streaming; machine learning

ACM Reference Format:
Tan Xu, Bo Han, and Feng Qian. 2019. Analyzing Viewport Prediction Under Different VR Interactions. In *The 15th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '19), December 9–12, 2019, Orlando, FL, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3359989.3365413

CoNEXT '19, December 9–12, 2019, Orlando, FL, USA

ACM ISBN 978-1-4503-6998-5/19/12…$15.00
https://doi.org/10.1145/3359989.3365413

1 INTRODUCTION

Powered by recent advances of networking and multimedia technologies, Internet video streaming has become unprecedentedly popular. Besides traditional videos, emerging multimedia content such as 360° VR videos [15, 24, 30, 36, 38], volumetric videos [35, 37], and real-time cloud gaming [11] are being or will be streamed to Internet users. Another exciting trend is that users can consume multimedia content on diverse devices, from PCs and smartphones to head-mounted devices (HMD). However, Internet video streaming is well known to be challenging, and streaming VR content incurs even more challenges due to VR's high bandwidth demand and viewers' stringent Quality of Experience (QoE) requirements. For example, users wearing VR headsets are highly vulnerable to video stall (rebuffering) events that may cause VR sickness [17].

Along with these challenges, VR streaming bears unique optimization opportunities. Among them, *viewport adaptation* has recently attracted significant attentions from the research community [15, 32, 36]. The basic idea is straightforward: instead of fetching a panoramic scene, the player downloads only the content to appear in the viewport, *i.e.,* the portion to be perceived by users. If properly applied, viewport adaptation can significantly reduce bandwidth consumption and player-side processing overhead, simply because less content is fetched and displayed. However, it is also highly challenging due to the "randomness" of viewport movement. Note that viewport adaptation is a general concept that is widely applicable to VR, AR, and MR (Mixed Reality).

In this paper, we investigate a critical component in any viewport adaptive system: viewport prediction (VP), *i.e.,* predicting what a viewer is about to consume in the near future. To accommodate the network latency and content processing delay, a networked VR/AR system typically needs to fetch and/or preprocess the content before it is displayed to viewers. When viewport adaptation is used, VP plays an important role in this process and its accuracy determines whether the proper content will be fetched. Downloading incorrect content that users do not end up viewing may lead to severe side effects: bandwidth/energy waste, lower perceived quality, video stall, and frame skip. Our previous work has demonstrated that even using a simple linear regression model for VP, a viewport adaptive VR video streaming system can improve the video quality level by up to 18× on WiFi and up to 4.9× on LTE, and save mobile data usage by up to 35% [38].

Our study begins with collecting real users' viewport trajectory data through IRB-approved user trials when they watch 10 360° videos, one of the most popular VR content types today. Compared to previous studies collecting human head movement data [15, 19, 33, 39], our datasets are unique in that they cover different interaction modes: watching 360° videos when wearing a head-mounted device (HMD), tilting a smartphone, and dragging the mouse on a PC. Overall, our datasets consist of 2,750 viewing sessions of 10

popular 360° videos that have been watched ~135 million times on YouTube. The datasets are contributed by a total number of 275 users, and have a total playback duration of 156 hours – much larger than most existing datasets. We intend to make the datasets and the data collection software publicly available. More details of this user study and our datasets are in §3.

Leveraging our unique datasets, we then conduct measurements to understand the viewport movement patterns of different interaction modes in §4. This is a problem under-explored by previous studies. Despite the viewing experience of 360° videos under these interaction methods (with 18 participants watching two videos) was investigated by an existing work [18], it is still not clear how they will affect the performance of viewport prediction. We find that when using HMDs, users tend to move their heads more smoothly with little fixation. Thus, it is relatively easy to predict their future viewports with the recent trajectory as input. On the other hand when using PCs, users have longer fixation (*i.e.,* do not change viewport for a longer period). However, when they actually move the viewport, the speed is faster than using HMDs and phones. This characteristic of PC users results in a more accurate prediction for a shorter window, but less accurate prediction for a longer window.

We next explore the core problem of VP by presenting three diverse machine learning techniques in §5: Linear Regression (LR), MultiLayer Perceptron regression (MLP), and Trajectory-based Crowd-sourced Deep Learning (TCDL, trained by viewport traces from multiple users), These algorithms are diverse in that they cover classic machine learning vs. deep learning, train from single vs. multiple users, and feedforward neural network vs. recurrent neural network. We also use a baseline, called *Static*, that assumes the viewport does not change during the prediction window.

We compare these VP methods using our datasets in §6. Our findings consist of the following: (1) regression models can accurately predict future viewports for short windows (*e.g.,* 0.1s), but its performance decreases for longer windows (*e.g.,* 2s); (2) trajectory-based deep learning model performs the best and is reliable across different interaction modes and prediction window sizes; and (3) when watching 360° videos using HMDs, the viewport prediction is more accurate than using smartphones and PCs.

Overall, the above results provide key insights on how to intelligently perform VP in networked VR systems. Although our study uses 360° videos as the content, we believe the findings are applicable to VR content in general. Our contributions can be summarized as follows.

- We contribute to the community large datasets of 360° video viewport trajectories.
- We measure viewport movement patterns under different interaction modes.
- We conduct comprehensive analysis to understand the VP accuracy and performance of diverse machine learning algorithms.

2 BACKGROUND AND RELATED WORK

360° videos allow users to freely change their viewport during playback, leading to unique, immersive viewing experience. They have become increasingly popular on commercial platforms such as YouTube and Facebook [7, 9]. As of today, almost all commercial platforms employ a *monolithic* approach for streaming 360° videos: the client player fetches the entire panoramic scene from the server, but displays to the user only a small portion within the viewport (typically around 1/4 [15, 36]). This approach is simple, but incurs high bandwidth waste since most of the downloaded content is not consumed by the viewer.

Recently, the research community has proposed numerous improvements to the above monolithic approach. Most of the improvements share on aspect in common: they apply the concept of *viewport adaptation*: instead of fetching the panoramic content, the client strategically downloads only the portion within the user's viewport, or fetches such regions at a high resolution and the remaining at lower resolutions. To realize this, the most popular approach is called *tiling*, where each panoramic video chunk (*i.e.,* a small video file with the duration of a few seconds) is further spatially segmented into *tiles* to allow the client to fetch a small region [24, 38]; alternatively, the server can also prepare multiple versions with different high-resolution regions for each chunk [20, 41].

Regardless of which algorithm to use, any viewport-adaptive approach requires *predicting the user's future viewport* as a key building block in order to fetch the content in a timely manner, in particular over wireless networks where bandwidth is a scarce resource. In the literature, most work on 360° video streaming do not consider viewport prediction (VP) [22], or use very simple prediction methods such as linear regression [24] and straightforward classic machine learning [16]. Only a very limited number of studies focus on VP itself [21, 27]. Within them, Fan *et al.* [21] propose to leverage both motion data from sensors and video content features (*e.g.,* image saliency maps) for predicting future viewports. For a given segmentation scheme with a tile size of 30° ×30°, Hou *et al.* [27] directly predict the tiles overlapping with a future viewport using a deep learning model.

Compared to the above work on VP, our study advances the state-of-the-art in two aspects. First, instead of focusing on a single interaction type, we compare VP over three ways of watching 360° videos: wearing a VR headset, using bare smartphones, and through PCs/laptops. Second, leveraging our unique datasets, we comprehensively investigate the accuracy of different VP methods, from classic machine learning to sophisticated deep learning based on viewport trajectory.

3 DATASETS

This section describes our datasets. We first select ten popular YouTube 360° videos according to their view counts as listed in Table 1. As of June 2019, these videos have been viewed ~135 million times in total. Regarding the content, they belong to diverse genres including scenery, sports, movie, performance, *etc.* The duration of the videos ranges from 2 to 5 minutes. We download them from YouTube at their highest quality (4K resolution) with the average encoded bitrate being 16.6 Mbps. All videos are encoded using Equirectangular projection.

We next conduct IRB-approved user studies to collect real users' viewport movement traces when watching these 10 videos. We notice that in the literature, there already exist several studies that collect head movement data from viewers when they watch 360°

ID	Category	Duration (s)	Bitrate (Mbps)	# Views (6/2019)
1	Animal [3]	169	12.9	2.91M
2	Film [1]	293	15.9	3.88M
3	Diving [2]	134	15.0	19.98M
4	Roller coaster [5]	117	17.1	53.49M
5	Scenery [8]	242	16.0	5.67M
6	Driving [4]	181	21.5	5.59M
7	Documentary [6]	204	15.1	3.38M
8	Performance [13]	265	15.8	9.64M
9	Animation [10]	203	17.8	2.10M
10	Skydiving [12]	233	18.4	28.21M

Table 1: 10 videos used in our study.

videos with a HMD [15, 19, 33, 39]. Our data collection methodology differs in a key aspect where we consider *different interaction methods*. Specifically, Table 2 lists three datasets. DS-HMD consists of head movement data of 130 users when they watch the 10 videos wearing a Samsung Gear VR headset with a smartphone plugged in. Viewers can simply adjust the viewport by moving their heads. Note that almost all prior studies focused on this type of interaction. DS-Phone instead captures 54 users' viewport trajectories as they watch the 10 videos on a bare smartphone (Samsung Galaxy S8). The users hold and move the phone, either by hand or by moving the body, to change the viewport. DS-PC records 91 users' viewport trajectories as they watch the videos on their PCs (either desktop or laptop). Changing the viewport is realized by dragging the mouse or swiping on the laptop's touchpad. All popular browser-based 360° video players such as those of YouTube and Facebook use this type of interaction on PCs.

To collect the above data, we recruit voluntary participants from a large U.S. university. A limitation of our study is that most of our participants are university students. Nevertheless, we have tried our best to increase the diversity of users. Specifically, we widely disseminate recruiting materials so that the participants come from more than 10 departments/schools as well as all education levels (from freshman undergraduate to senior Ph.D. students); we have also managed to have a non-trivial fraction of faculty and staff members join our study. In addition, we ensure that the participants are not overwhelmingly dominated by a particular gender or by people who have or have not watched 360° videos before, as shown in Table 2.

For each dataset, we develop its corresponding data collection software. For DS-HMD and DS-Phone, the data collector runs as a standalone Android app, while for DS-PC, the software is developed using Javascript and executes in a Chrome browser. For all interaction methods, the collectors record time-series data of each user's viewport position in latitude and longitude. The viewport size is fixed to 100° × 90°. Overall, the datasets consist of 2,750 viewing sessions of 360° videos from these users, with a total playback duration of 156 hours. To the best of our knowledge, this is the first effort of collecting 360° video viewport trajectories using different interaction methods.

4 COMPARING INTERACTION METHODS

To characterize the viewport movement for each interaction method, we measure the viewport moving speed, duration of idle period

Name	# Users	% Female	% Novice	% Undergrad.
DS-HMD	130	42%	48%	35%
DS-Phone	54	36%	32%	32%
DS-PC	91	51%	26%	58%

Table 2: A summary of our datasets. Novice users are those who have not watched 360° videos before.

(*i.e.,* static viewport) and plot the heatmaps of viewport trajectories for these videos.

For the speed of viewport movement (°/second), we plot the cumulative distribution function (CDF) for each interaction method in the vertical direction (latitude) and horizontal direction (longitude) in Figure 1 and Figure 2, respectively. When calculating these speeds, if viewers move the viewport along the same direction, we calculate the speed every 1 second. If viewers change their moving direction, we calculate the speed for the previous period with the same movement direction. In order to clearly demonstrate the difference between these interaction methods, we exclude the speeds lower than 3°/second, which are considered as static, and those higher than 180°/second for both latitude and longitude. These high speed movements may occur when a participant suddenly changes the viewing direction. There are two observations from these figures. First, when watching 360° videos using PCs, the viewport movement speed is much higher than that for using HMDs and phones. For PC users, the median latitude and longitude speed is 14.8°/second and 27.6°/second, respectively. Second, the viewport movement speed for phone users is only slightly higher than that for HMD users. For HMD users, the mean latitude and longitude speed is 6.19°/second and 5.20°/second, respectively.

We then compare the duration of "static" viewport period for these interaction methods and plot the CDFs in Figure 3. We say the viewport of a user is static if the moving speed is lower than 3°/second for both latitude and longitude. To clearly demonstrate the difference between these interaction methods, we plot the CDFs of these durations longer than 3 seconds. While PC users have the fastest viewport movement speed among the three, their duration of static viewport period (7.5 seconds) is also higher than the other two interaction methods. The reason is that although PC users tend to not move their mouse while watching 360° videos (no change of viewport), when they do change the viewport, they can move faster than HMD and phone users. It is worth mentioning that the distribution of this duration for PC users has a long tail (not shown in the figure) with the top 10 values higher than 100 seconds. There is no noticeable difference for the duration distributions between HMD and phone users.

To further illustrate the viewport movement patterns of these three methods in a finer granularity, we plot the heatmaps for these videos using the viewport trajectories of 30 randomly selected users. We want to compare these interaction methods based on the same number of users (limited by the size of DS-Phone), while applying the sampling to further reduce data bias in users. We show the heatmaps for a representative video in Figure 4, which projects the trajectories recorded in latitude and longitude on a canvas (3840 × 2048 pixels), the same size as the 360° video frames after Equirectangular projection. For HMD users, their viewports have a focused area. For phone users, the consumed portions are much larger than the other two methods, mainly because it is

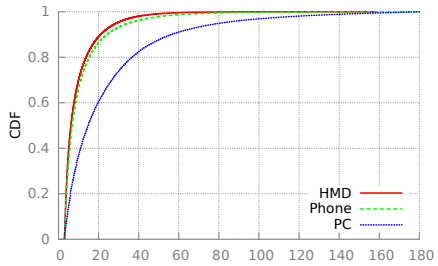


Figure 1: Latitude movement speed (degrees/second).

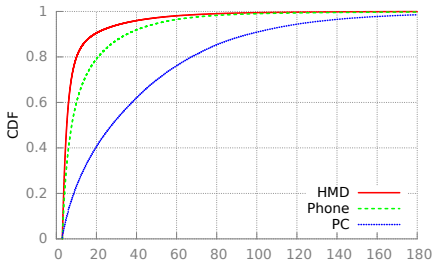


Figure 2: Longitude movement speed (degrees/second).

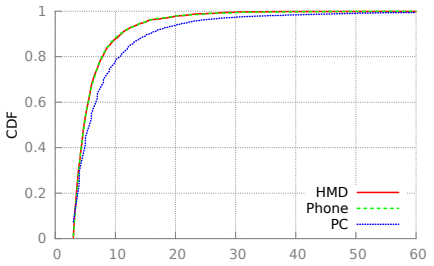
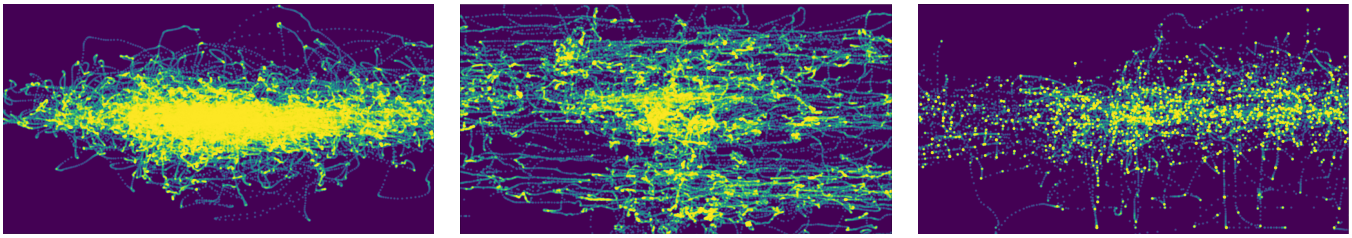


Figure 3: Duration of idle-period (seconds).



(a) HMD **(b) Phone** **(c) PC**

Figure 4: Heatmap created from the trajectory of 30 randomly selected users for video #3.

relatively easy to move the phone around when watching a 360° video on a smartphone. For PC users, their viewports are scattered due to the infrequent movement of their viewports. These above differences among the interaction methods, as reflected from the data we collect, affect the viewport prediction modeling techniques and their accuracy, which we will reveal in next section.

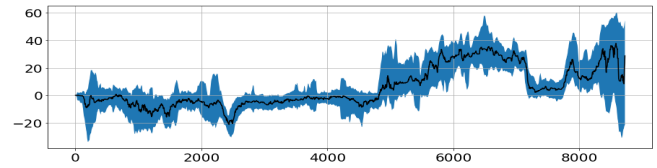
5 VP METHODS

As described in §2, viewport prediction (VP) is an essential component of any viewport-adaptive VR/AR content delivery scheme. Accurate VP can help effectively reduce the bandwidth consumption and video stall. In this section, we present three diverse VP methods and their training procedures. Because of the sequential nature of the viewport trajectory data, which is a time series of viewport-center positions recorded by latitude and longitude at a sampling rate of 30Hz, we investigate regression models [24, 32, 36, 38] and recurrent neural network (RNN) [21, 27] by following existing work. Given the recent viewports in a history window *hw*, the goal is to predict the next viewports in a prediction window *pw*. We use *Static* as a baseline, which keeps the last observed viewport unchanged in the prediction window.

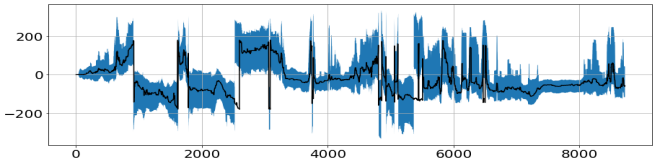
Linear Regression (LR) uses a linear model to approximate a user's short-term viewport movement. We use each viewport position (latitude, longitude) as dependent variable and their relative order in the sequence to the 1*st* point in *hw* as independent variable. At runtime, we repeatedly fit a linear model from the recent viewport trajectories falling into *hw*, and use the model to predict future viewports in *pw*. The latitude and longitude are separately trained and predicted. LR is simple, fast, and reasonably accurate. It is thus used in several viewport-adaptive systems [24, 38].

MultiLayer Perceptron Regressor (MLP) is a typical non-linear model that employs a feedforward neural network to perform prediction. When constructing the MLP, we empirically choose hyperbolic tangent function [28] for activation and Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) [31] for optimization. Given the simplicity of our data (with only one dimensional independent variable), we decide to use a single hidden layer with three neurons so that the model could be trained and predict fast. Similar to LR, we perform online training and prediction whose time windows are determined by *hw* and *pw*, respectively.

Trajectory-based Crowd-sourced Deep Learning (TCDL). Given the numerous applications and effectiveness of RNN in processing sequential data such as text and speech, we apply a classic RNN architecture, Long Short-Term Memory (LSTM) [23, 26] to VP. The key to LSTM is the multiplicate gates, which allows its memory cells to store and access information over long periods of time. Therefore LSTM can better avoid the vanishing and exploding gradient problem [14]. Compared with other methods such as Markov chains and hidden Markov models, LSTM does not presume Markov assumption and thus can better exploit the potential patterns for modeling sequential data [34]. Furthermore, LSTM has highly rich representation and model capacity compared with LR and MLP. This allows LSTM to discover deeper viewport movement patterns, *e.g.,* long-term trend and seasonality, cross-viewer interests, or content driven movement. Therefore, when training LSTM, we use multple pairs of *hw* and *pw* trajectories sampled from various conditions to fit the model, *e.g.,* from multiple users and at any moment of a video. As a result, the model can be trained completely offline and predict with unseen *hw* inputs of trajectory. Contrarily, LR



(a) Latitude



(b) Longitude

Figure 5: Cross-viewer trajectory commonality for video #2.

and MLP have to work in an online fashion, which is still practical because of their lightweight and fast training.

For better generalization to unseen data, our constructed LSTM architecture starts with a Subtraction layer that performs 1*st* point normalization after the input layer. It then employs one layer of LSTM with 64 neurons, and an additional Add layer that denormalizes the values before output. We notice that there are several hyperparameters such as the number of LSTM layers and neurons that could be fine-tuned for achieving better performance. We leave this as our future work. When training such a model, we choose Mean Absolute Error (MAE) as the loss function, which is the same metric used in evaluation, and Adaptive Moment Estimation (ADAM [29]) as the optimizer.

With regard to the training data for LSTM, we randomly sample trajectories from n users, and select $n \in [3, 30]$ to investigate the impact of size n. Then, we separate user trajectories by videos. The rationale of training from crowd-sourced video specific trajectories is that, despite the randomness of users' viewport movement, different users may oftentimes change viewing directions similarly as attracted by the same video content. This is illustrated in Figure 5, for which the X axis is the frame index of a random representative video and the Y axis is the viewport position in latitude and longitude. The black solid line is the median position cross 30 randomly sampled viewers, and the blue band denotes the standard deviation. The band appears to be narrow in many frames, indicating the consistency of the viewers' viewport positions (in particular for the latitude). Another way to formulate training data is per user or user group but across multiple videos. However, in practice we are more interested in learning video-specific models than user-specific models, as it is challenging to apply the latter to a new user.

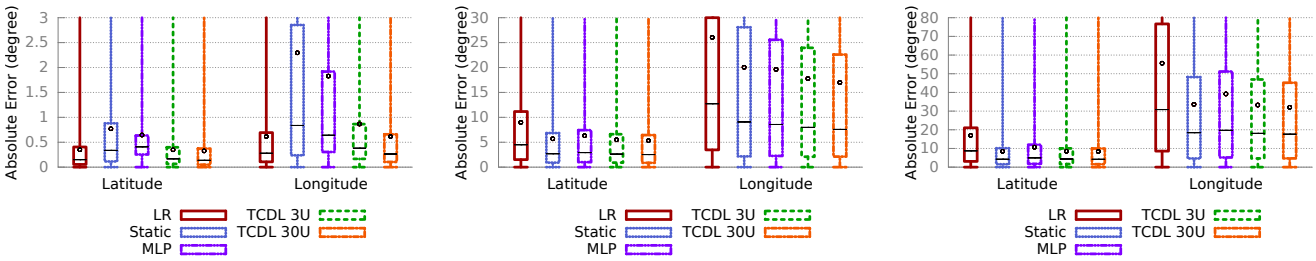
6 EVALUATION

To evaluate the performance of the above proposed VP schemes, we conduct a 2-fold cross validation on all three datasets. We check Absolute Error (AE) for the accuracy, which is calculated from each prediction against users' real trajectory. One experiment setup decision we need to make is the size of pw. We highlight 3 options: $0.1s$, $1.0s$, and $2.0s$ (to example short, intermediate, and long pw), with each we vary hw from $0.05s$, $0.6s$, and $1.0s$ correspondingly. According to a previous work, this setup can yield the best performance for LR [38]. Given a user's viewport trajectory, we slide a time window of hw and use its next pw for training and test.

We first compare VP methods. We highlight their performance on DS-HMD as shown in Figure 6. We have the following observations from these results. (1) Regression models can provide accurate predictions for short pw, however, their accuracy decreases faster than TCDL when pw increases. Depends on network condition and video quality, if it takes short time to stream and process a 360° video tile, which means we only need to make a short-term decision about which tile to pre-fetch, a regression model may be a good choice because it is lightweight, easy to implement, and achieves a good accuracy for short pw viewport prediction. However, if it takes long time to stream and process a tile, which means a long-term decision is needed, then TCDL can provide a more accurate prediction. (2) Static baseline performs surprisingly good, especially in long pw. The reason may be that the randomness in viewport movement weakens the suggestiveness of recent trajectory. This provides a strong backup plan when machine learning based VP fails, as it requires no computing or other input than the last available viewport position. (3) TCDL consistently outperforms the baseline and regression methods for all pw. More training users only helps improve the accuracy marginally. This is important, because for a new video, it means TCDL model can be quickly trained with even a few users' trajectories.

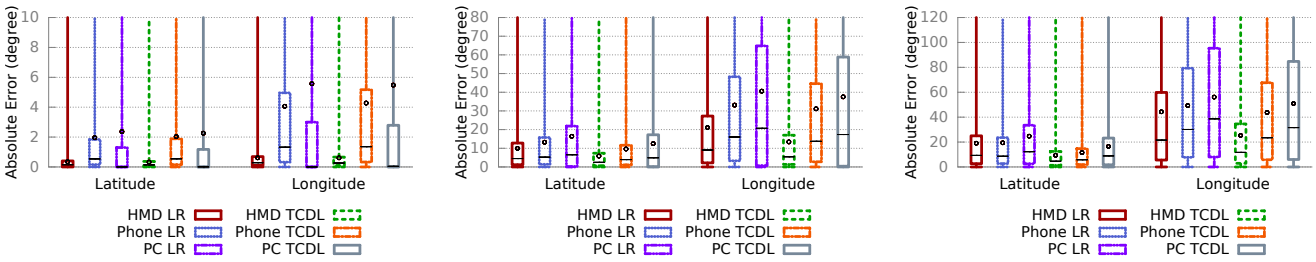
Next, we evaluate the impact of different interaction methods on the accuracy of viewport prediction. Figure 7 shows the AE boxplot of the three datasets for different pw, with latitude and longitude results displayed separately. Note that only LR and TCDL (trained with 30 users) results are shown here to clarify the figures. We have the following observations from these results. (1) For all three datasets, the prediction accuracy decreases when pw increases, which confirms long term VP is a more difficult problem, even for the PC users who have relative longer idle period. On the other hand, both LR and TCDL can achieve nearly perfect predictions for short pw - around 1 degree AE when $pw = 0.1s$. (2) In general, the prediction error doubles in longitude compared to latitude. This is due to the doubled freedom of movement in horizontal direction $\in [-180°, +180°]$ vs. vertical direction $\in [-90°, +90°]$. (3) The prediction accuracy of HMD users is better than phone users, and the accuracy of phone users is better than PC users, except for short pw, where the viewport of PC users can be predicted more accurately. This can be explained by their different movement patterns. PC users have relative longer idle period. Thus, for short pw, it is easier to predict the viewport. HMD and phone users move more smoothly. Thus, their viewport is more predictable in longer pw. (4) TCDL always yields better prediction accuracy than LR across all three interaction methods for all pw.

When evaluate the performance on each individual video, we notice sometime the improvement of TCDL over other methods including the simple Static baseline is moderate. Specifically, we find for two videos (#4 and #6), TCDL has only marginal gain in both short and long pw. We then analyze these two videos for more insight, and find that they both can be characterized as driving content with high motion content at both sides of a driving trail. When watching these videos, most users' viewports consistently center around the driving trail with almost no movement. A random 30



(a) hw = 0.05s, pw = 0.1s. (b) hw = 0.6s, pw = 1s. (c) hw = 1s, pw = 2s.

Figure 6: Comparisons of VP methods using HMD dataset.



(a) hw = 0.05s, pw = 0.1s. (b) hw = 0.6s, pw = 1s (c) hw = 1s, pw = 2s.

Figure 7: Comparisons of VP for three interaction modes.

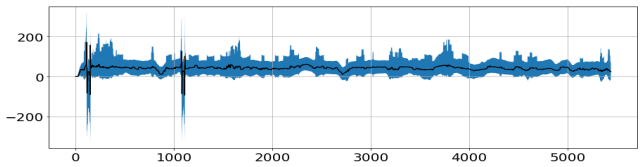


Figure 8: Cross-viewer longitude trajectory for videos #6.

users' longitude trajectories for video #6 is shown in Figure 8. Thus, it results in high prediction accuracy for all VP models with little difference from each other. In another thread of work, we investigate whether content features can help in viewport prediction by adding them into our TCDL model [40]. However, we can only successfully improve the VP accuracy over the trajectory-only models as proposed here for some videos, especially videos characterized with apparent central objects moving in slow motion, which can surely drive users' attention, *e.g.,* a walking elephant in video #1 or a swimming diver in video #3.

7 CONCLUSION AND FUTURE WORK

Viewport prediction plays an important role in networked VR systems by dictating what content to fetch. As demonstrated by existing work, even with a simple LR for VP, a viewport-adaptive 360° video streaming system implemented for mobile devices could yield high bandwidth savings and significant improvement in user QoE [25, 38]. In this study, with the datasets we created from 275 users (watching 10 videos for a total playback duration of 156 hours), we analyze people's viewport movement when watching 360° video under three VR interaction methods (PC, smartphone and HMD) thoroughly. We then explore various machine learning models to more accurately predict future viewports. Our key findings include but not limited to: (1) when using PCs for watching 360° videos, users tend to be in the idle state without changing the viewport for a longer duration than using HMDs and phones; (2) trajectory-based deep learning scheme performs best among the evaluated approaches, regardless of the interaction methods and prediction window sizes; and (3) the viewport prediction is more accurate for HMD users than phone and PC users mainly due to their smooth viewport movement pattern.

In our future work, we plan to fine-tune models studied in this work and implement them into a real viewport-adaptive 360° video streaming system. In addition, we would like to explore and integrate other modalities such as user preference, video content and spatial audio (*e.g.,* the sound source in a 3D stereo audio system) for more accurate and long term viewport prediction. Furthermore, we plan to extend studies in this work to investigate 6DoF (degree-of-freedom) viewport prediction for the emerging volumetric video streaming, which is a more challenging problem but can definitely benefit from accurate viewport prediction.

ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers and our shepherd Zubair Shafiq for their valuable comments. We would also like to thank the voluntary users who participated in our user study. Feng Qian's research was supported in part by NSF Award #1915122 and a Google Faculty Award.

REFERENCES

[1] 360 Google Spotlight Story: Help . https://www.youtube.com/watch?v=G-XZhKqQAHU.

[2] 360° Great Hammerhead Shark Encounter . https://www.youtube.com/watch?v=rG4jSz_2HDY.

[3] Elephants on the Brink . https://www.youtube.com/watch?v=2bpICIClAIg.

[4] GT-R Drives First EVER 360 VR lap . https://www.youtube.com/watch?v=LD4XfM2TZ2k.

[5] Mega Coaster: Get Ready for the Drop . https://www.youtube.com/watch?v=-xNN-bJQ4vI.

[6] Pony Stable Playhouse for the Currys . https://www.youtube.com/watch?v=MWg1kjMmr3k.

[7] Under the hood: Building 360 video . https://code.facebook.com/posts/1638767863078802.

[8] Visit Hamilton Island in 360° Virtual Reality with Qantas . https://www.youtube.com/watch?v=lJype_TafRk.

[9] YouTube live in 360 degrees encoder settings . https://support.google.com/youtube/answer/6396222.

[10] Feel wimbledon with andy murray. https://www.youtube.com/watch?v=Krl6U15OERo.

[11] Google Stadia. https://en.wikipedia.org/wiki/Google_Stadia.

[12] Skydive in 360° virtual reality via gopro. https://www.youtube.com/watch?v=S5XXsRuMPIU.

[13] Tomorrowland 2014 | 360 degrees of madness. https://www.youtube.com/watch?v=j81DDY4nvos.

[14] Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and De Freitas, N. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems* (2016), pp. 3981–3989.

[15] Bao, Y., Wu, H., Zhang, T., Ramli, A. A., and Liu, X. Shooting a moving target: Motion-prediction-based transmission for 360-degree videos. In *Proceedings of Big Data 2016* (2016), IEEE, pp. 1161–1170.

[16] Bao, Y., Zhang, T., Pande, A., Wu, H., and Liu, X. Motion-Prediction-Based Multicast for 360-Degree Video Transmissions. In *Proceedings of SECON 2017* (2017), IEEE, pp. 1–9.

[17] Boos, K., Chu, D., and Cuervo, E. Flashback: Immersive virtual reality on mobile devices via rendering memoization. In *Proceedings of MobiSys 2016* (2016), ACM, pp. 291–304.

[18] Broeck, M. V. d., Kawsar, F., and Schö ning, J. It's All Around You: Exploring 360° Video Viewing Experiences on Mobile Devices. In *Proceedings of MM 2017* (2017), ACM, pp. 762–768.

[19] Corbillon, X., De Simone, F., and Simon, G. 360-Degree Video Head Movement Dataset. In *Proceedings of MMSys 2017* (2017), ACM.

[20] Corbillon, X., Simon, G., Devlic, A., and Chakareski, J. Viewport-adaptive navigable 360-degree video delivery. In *Proceedings of ICC 2017* (2017), IEEE.

[21] Fan, C.-L., Lee, J., Lo, W.-C., Huang, C.-Y., Chen, K.-T., and Hsu, C.-H. Fixation Prediction for 360 Video Streaming in Head-Mounted Virtual Reality. In *Proceedings of the Workshop on Network and Operating Systems Support for Digital Audio and Video* (2017), ACM, pp. 67–72.

[22] Graf, M., Timmerer, C., and Mueller, C. Towards bandwidth efficient adaptive streaming of omnidirectional video over HTTP: Design, implementation, and evaluation. In *Proceedings of MMSys 2017* (2017), ACM, pp. 261–271.

[23] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems 28*, 10 (2016), 2222–2232.

[24] He, J., Qureshi, M. A., Qiu, L., Li, J., Li, F., and Han, L. Rubiks: Practical 360-Degree Streaming for Smartphones. In *Proceedings of MobiSys 2018* (2018), ACM.

[25] He, J., Qureshi, M. A., Qiu, L., Li, J., Li, F., and Han, L. Rubiks: Practical 360-degree streaming for smartphones. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services* (New York, NY, USA, 2018), MobiSys '18, ACM, pp. 482–494.

[26] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[27] Hou, X., Dey, S., Zhang, J., and Budagavi, M. Predictive View Generation to Enable Mobile 360-degree and VR Experiences. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network* (2018), ACM.

[28] Karlik, B., and Vehbi, A. Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks. *International Journal of Artificial Intelligence and Expert Systems 1*, 4 (2010), 111–122.

[29] Kingma, D. P., and Ba, J. Adam: A Method for Stochastic Optimization. In *Proceedings of International Conference on Learning Representations* (2015).

[30] Lai, Z., Hu, Y. C., Cui, Y., Sun, L., and Dai, N. Furion: Engineering high-quality immersive virtual reality on today's mobile devices. In *Proceedings of MobiCom 2017* (2017), ACM, pp. 409–421.

[31] Liu, D. C., and Nocedal, J. On the limited memory BFGS method for large scale optimization. *Mathematical Programming 45*, 1–3 (1989), 503–528.

[32] Liu, X., Xiao, Q., Gopalakrishnan, V., Han, B., Qian, F., and Varvello, M. 360 Innovations for Panoramic Video Streaming . In *Proceedings of HotNets 2017* (2017), ACM.

[33] Lo, W.-C., Fan, C.-L., Lee, J., Huang, C.-Y., Chen, K.-T., and Hsu, C.-H. 360 Video Viewing Dataset in Head-Mounted Virtual Reality. In *Proceedings of MMSys 2017* (2017), ACM, pp. 211–216.

[34] Panzner, M., and Cimiano, P. Comparing hidden markov models and long short term memory neural networks for learning action representations. In *International Workshop on Machine Learning, Optimization and Big Data* (2016), Springer, pp. 94–105.

[35] Park, J., Chou, P. A., and Hwang, J.-N. Rate-Utility Optimized Streaming of Volumetric Media for Augmented Reality. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems 9*, 1 (2019), 149–162.

[36] Qian, F., Han, B., Ji, L., and Gopalakrishnan, V. Optimizing 360 video delivery over cellular networks. In *Proceedings of the Workshop on All Things Cellular: Operations, Applications and Challenges* (2016), ACM, pp. 1–6.

[37] Qian, F., Han, B., Pair, J., and Gopalakrishnan, V. Toward practical volumetric video streaming on commodity smartphones. In *Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications* (2019), ACM, pp. 135–140.

[38] Qian, F., Han, B., Xiao, Q., and Gopalakrishnan, V. Flare: Practical Viewport-Adaptive 360-Degree Video Streaming for Mobile Devices. In *Proceedings of MobiCom 2018* (2018), ACM.

[39] Wu, C., Tan, Z., Wang, Z., and Yang, S. A Dataset for Exploring User Behaviors in VR Spherical Video Streaming. In *Proceedings of MMSys 2017* (2017), ACM, pp. 193–198.

[40] Xu, T., Fan, Q., and Han, B. Content assisted viewport prediction for panoramic video streaming. In *Workshop on Computer Vision for AR/VR* (2019), CVPR '19, IEEE.

[41] Zhou, C., Li, Z., and Liu, Y. A Measurement Study of Oculus 360 Degree Video Streaming. In *Proceedings of MMSys 2017* (2017), ACM.