Viewing 360 Degree Videos: Motion Prediction and Bandwidth Optimization

Yanan Bao, Huasen Wu, Albara Ah Ramli, Bradley Wang and Xin Liu
Department of Computer Science, University of California, Davis, CA 95616, USA
{ynbao, hswu, arramli, radwang, xinliu}@ucdavis.edu

***Abstract*—360-degree video transmission consumes 4∼6x the bandwidth of a regular video, and thus imposes significant challenges to networks. To address this challenge, in this paper, we propose a motion-prediction-based transmission mechanism that matches network video transmission to viewer needs. Ideally, if viewer motion is perfectly known in advance, we could reduce bandwidth consumption by 80%. Practically, however, we have to address the random nature of viewer motion, in order to guarantee the quality of the viewing experience. Based on our experimental study of viewer motion (comprising 16 video clips and over 150 subjects), we propose a machine learning mechanism that predicts viewer motion. Based on such predictions, we propose a partial-content-transmission mechanism that reduces the overall bandwidth consumption while providing probabilistic performance guarantees. Real-trace-based evaluations show that the proposed scheme significantly reduces bandwidth consumption with negligible performance degradation. For example, given a failure ratio of 0.1%, we can reduce bandwidth consumption by more than 40%.**

I. Introduction

In virtual reality (VR), viewers typically watch 360-degree videos using head-mounted displays (HMDs). When watching a 360-degree video, at any given time, a viewer is facing a certain direction. Thus, the HMD needs to render and display only the content in this viewing direction, which is typically 20% of the whole sphere. Therefore, if the viewer's viewpoint can be predicted well, she needs to receive only a portion of the content, greatly reducing the bandwidth consumption.

We first built a testbed to collect 3D motion data using HMDs. Wearing an HMD, a viewer's motion has three degrees of freedom (pitch, yaw, and roll), as illustrated in Fig. 1. Our experiment collected motion data in the three dimensions, based on 16 clips of 360-degree video and 153 viewers. In analyzing the collected data, we found that viewer motion had strong short-term auto-correlations in all three dimensions. Using regression techniques, we could predict viewer motions with reasonable accuracy on a time scale of 100-500 ms, as shown in Sec. III.

However, due to the random nature of viewer motion, motion prediction is prone to error. Given such error-prone prediction, we need to transmit a larger area than the field of view (FOV, shown in Fig. 2), in order to guarantee viewing quality. Based on the motion prediction, we design a transmission scheme to decide which portion of the content to transmit to the viewer. The objective is to minimize the required bandwidth while upholding viewing quality guarantees.

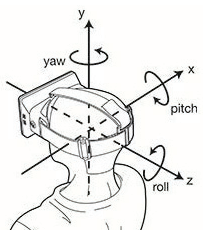

Fig. 1. The three angles [1]

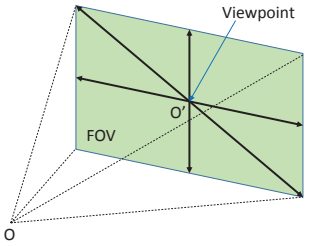

Fig. 2. Field of view

II. Experiment Setup and Data Collection

A. Hardware and Software

We use the Oculus DK2 as the hardware, Oculus Runtime 7.0 as the hardware driver and *Color Eyes* as the video player. We developed a software to play video clips automatically to viewers. Subjects sit on a chair that can rotate horizontally 360°. The HMD of Oculus DK2 is connected to a PC with a cable.

B. 360 Video Content and Motion Measurement

We downloaded 16 clips of 360-degree video from Youtube and cut each of them into 30 seconds. The video clips and sample motion data can be found at http://360videoexp.com/. When a subject is watching a video, his or her motion is recorded and logged. As usual, the motion includes 3 degrees of freedom, pitch, yaw and roll (i.e., X, Y, and Z angles). When wearing an HMD, the viewer's initial position defines the zero degree for pitch, yaw and roll. Each dimension is denoted by an angle ($-180°$ to $180°$).

C. Subjects

In total, 153 volunteers joined the experiment: 35 of them watched all 16 video clips and 118 of them watched 3∼5 randomly selected video clips.

D. Data Preprocessing

The overall collected data include 985 views from the subjects. Our software collected 7∼9 samples per second, and the intervals between two samples were slightly random. To facilitate the following study, we generate uniformly 10 samples per second from the raw data using linear interpolation. After interpolation, we have in total about 295500 samples.

TABLE I
THE ERROR OF Y ANGLE PREDICTION (VALUES IN DEGREES)

T_r(s)	0.1	0.2	0.3	0.4	0.5
Naive (Mean)	2.58	5.09	7.53	9.89	12.16
Naive (RMSE)	4.71	9.10	13.23	17.06	20.58
Naive (99th)	18.24	35.24	50.85	65.14	77.89
Naive (99.9th)	32.59	62.75	88.90	112.90	130.25
NN (Mean)	0.92	2.44	4.33	6.33	8.40
NN (RMSE)	1.92	4.52	7.77	11.09	14.48
NN (99th)	6.54	17.25	30.54	44.03	57.59
NN (99.9th)	14.34	35.16	61.02	84.46	107.14

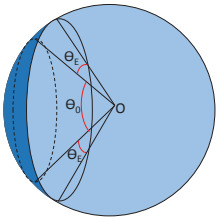

Fig. 3. The transmitted round

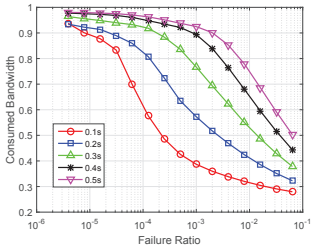

Fig. 4. Consumed bandwidth v.s. failure ratio

III. MOTION PREDICTION

By analyzing the collected motion traces, we found that viewer motion has strong temporal auto-correlation in a short period. This observation suggests that we can predict the future viewpoint based on existing samples.

Specifically, in this paper, we consider two models: Naive and neural networks (NN). The Naive model is the baseline model where we use the current angle as the value of the future angle. In the NN model, we use 3 layers and 5 hidden neurons, with samples in the past second as input. To address the periodicity issue of angle prediction, we first project the angles to a circle of unit radius, use the projected physical location on the circle for prediction, and then project the location back to the predicted angles.

Table I shows the test error for the prediction of the Y angle, which is harder to predict than the X angle. We use 50% of the data for training the model with the objective of minimizing the sum of square errors, and the other 50% for test. Table I shows that the further the prediction, the larger the error. Compared with naive prediction results, the neural network models achieve a better accuracy. For example, given a prediction window of 0.2s, all four indicators (mean, root-mean-square error (RMSE), 99th percentile and 99.9th percentile) improve by about 50%.

IV. MOTION-PREDICTION-BASED TRANSMISSIONS

Given a viewpoint prediction, we obtain the predicted FOV, which can be covered by a circle with beam angle θ_0. In order to guarantee the viewer experience, we add a margin with beam angle θ_E to the circle in order to obtain a larger circular area that we transmit. This transmitted circular area has a beam angle of $\theta_0 + 2\theta_E$.

Assume there are N frames in the videos being watched. To measure the viewer experience, we introduce an indicator I_i^f to denote whether the frame i is a failure or not, where $I_i^f = 1$ if frame i has a subset of pixels required by the viewer but not transmitted and $I_i^f = 0$ otherwise. The indicator I_i^f is decided by the following parameters: the prediction error e_i^x and e_i^y, the Z angle Z_i, and the beam angle of the transmitted circle $\theta_0 + 2\theta_E$. Therefore, we denote $I_i^f = f_f(e_i^x, e_i^y, Z_i, \theta_0 + 2\theta_E)$.

Given the failure ratio r_f as user experience constraint, we need to decide the optimal margin θ_E by solving the following optimization problem numerically.

$$\arg\min_{\theta_E} \quad \text{area}(\theta_0 + 2\theta_E); \tag{1}$$

$$s.t. \quad \frac{1}{N}\sum_{i=1}^{N} I_i^f \le r_f; \tag{2}$$

$$I_i^f = f_f(e_i^x, e_i^y, Z_i, \theta_0 + 2\theta_E). \tag{3}$$

Since the larger θ_E is, the more the bandwidth it is consumed, and the less the failure occurs, binary search can be used to obtain the optimal θ_E effectively.

V. PERFORMANCE EVALUATION

We evaluate the proposed algorithms based on the collected motion data. We run 10 iterations to obtain the average values. In each iteration, 50% of the data are used for training the prediction models; the other 50% are used to evaluate the bandwidth requirement under a given failure ratio constraint.

Figs. 4 shows the consumed bandwidth vs. failure ratio of our designed scheme, for a prediction window ranging from 0.1s to 0.5s. We can see that for a given failure ratio, the required bandwidth increases as the prediction window increases. For instance, given a failure ratio of 0.1%, 10-20% additional bandwidth is consumed when the prediction window grows from 0.1s to 0.2s, or from 0.2s to 0.3s. This is because when the prediction window T_r is smaller, we can obtain more accurate predictions that help us target the viewing area and reduce the required bandwidth. For example, given a prediction window of 0.2s, we can reduce the transmission bandwidth by more than 40% given a failure ratio of 0.1%.

VI. CONCLUSION

To address the significant bandwidth requirements of 360-degree videos, we propose a motion-prediction-based transmission scheme. First, based on our collected viewer motion data, we show that motion prediction is feasible within a 100-500 ms timeframe. Based on this observation, we develop regression models that predict the viewpoint, and design a partial content transmission scheme that guarantees the viewer experience. Our trace-driven simulation results show significant bandwidth reduction. For example, given a prediction window of 0.2s, our proposed scheme can reduce bandwidth consumption by more than 40% while guaranteeing a failure ratio of 0.1%.

VII. ACKNOWLEDGMENTS

This work was partially supported by NSF through grants CNS-1547461; CNS-1457060; CCF-1423542.

REFERENCES

[1] V. Oculus, "Oculus rift," *Available at http://www.oculusvr.com/rift*, 2015.