View Direction and Bandwidth Adaptive 360 Degree Video Streaming using a Two-Tier System

Fanyi Duanmu, Eymen Kurdoglu, Yong Liu and Yao Wang

New York University, Tandon School of Engineering, Brooklyn, NY 11201, USA
{fanyi.duanmu, eymen.kurdoglu, yongliu, yaowang} @nyu.edu

ABSTRACT

360 degree video compression and delivery is one of the key components of virtual reality (VR) applications. In such applications, the users may freely control and navigate the captured 3D environment from any viewing direction. Given that only a small portion of the entire video is watched at any time, fetching the entire 360 degree raw video is therefore unnecessary and bandwidth-consuming. In this work, a novel two-tier 360 degree video streaming scheme is proposed to accommodate the dynamics in both network bandwidth and viewing direction. Based on the real-trace driven simulations, we demonstrate that the proposed framework can significantly outperform conventional 360 video streaming schemes.

Index Terms—*360 Degree Video, Virtual Reality, Video Streaming, Cellular Network.*

I. INTRODUCTION

In recent years, Virtual Reality (VR) technologies have been rapidly commercialized with tremendous applications continuously evolving to meet the market demands and consumer expectations, including the immersive cinema, VR gaming, education and training, tele-presence, social media, healthcare, etc. The streaming of 360 degree videos therefore becomes an important supporting technology for VR realization, to provide users with 360 degree views and unique immersive experience. Such video services are very popular in recent years and made available on several major video platforms such as YouTube [1], Facebook [2], etc. However, the current delivery solutions of 360 videos are mainly derived from the traditional videos. Namely, the video players always fetch the entire 360 videos without considering the fact that viewers only watch a small portion of the entire 360 degree scene at a time.

There are several solutions proposed in recent years to address 360 degree video streaming. In [3], a set of tile-based encoding and streaming solutions are proposed, including scalable coding scheme and simulcast coding scheme. Basically, video tiles that cover the entire 360 scene are coded in multiple rates. Depending on the Field of View (FOV), the tiles within or close to predicted FOV is fetched with higher bitrate while the tiles far away from the predicted FOV are fetched with lower bitrate. In [4] and [5], Facebook's cube-map and pyramid projection and encoding schemes are described to address the on-demand 360 video streaming, with 25% and 80% compression gains reported. In [6], a region-adaptive smoothing scheme is proposed to reduce the bitrate spent within the polar regions through Gaussian filtering. A 20% bitrate reduction is reported with unnoticeable perceptual quality degradation. In [7], a view prediction based framework is proposed by only fetching the video portions desirable to the end user to reduce the bandwidth (BW) consumption. A dynamic video chunk adaptation scheme is implemented to adjust the tile area based on the view prediction accuracy. An estimated 80% peak rate reduction is reported without considering the coding efficiency loss due to video tiling and BW variations.

Inspired by the previous works, we propose a novel two-tier system to code and stream 360 degree videos, to both improve the bandwidth utilization and handling the network and viewing angle variations efficiently. The sequel of this paper is structured as follows. In Section 2, our proposed framework is illustrated and our design principles are discussed. In Section 3, we describe our experiment settings in details against two benchmark configurations. In Section 4, the experimental results are provided to demonstrate the potentials of our proposed framework. Section 5 concludes the paper with suggested future work summarized.

II. TWO-TIER FRAMEWORK FOR 360 VIDEO CODING AND STREAMING

Our proposed framework consists of two tiers of video representations as illustrated in Fig 1. For simplicity, We partition a 360 video into fixed-length (e.g. 1 second) video segments. Each segment is coded as a base-tier (BT) chunk, and multiple enhancement-tier (ET) chunks.

The BT chunks represent the entire 360 view at a low bit rate and are pre-fetched in a long display buffer to smooth the network jitters effectively and guarantee that any desired FOV can be rendered with minimum stalls.

The ET chunks represent the full 360 view range with multiple overlapping view coverages (VCs). For the same VC, multiple chunks are generated with different bitrates. At any new chunk request time, the receiver will predict the user's VC (defined by the center and width of the user's desired view) for the next segment, and only requests one of the chunks for the VC that covers the predicted FOV. The rate of the chunk is determined by the estimated network bandwidth. As studied in [7], the long-term head motion prediction is very difficult. The prediction accuracy drops drastically from 92% down to 71% when the prediction time increases from 1 second ahead to 2 second ahead. Therefore,

978-1-4673-6853-7/17/$31.00 ©2017 IEEE

the ET chunks in our system are pre-fetched in a very shallow buffer (i.e., 1 second ahead) such that the delivered VC mostly coincides with the actual user FOV. When users view directions are predicted accurately and ET chunks are received successfully, the client video player can combine the ET chunk with the pre-fetched BT chunk for an enhanced quality in the user's VC. Even when the view prediction fails (e.g., due to the unexpected head motion) or when the requested ET chunk does not arrive in time, the client can still render the desired view with low quality from the pre-fetched BT.

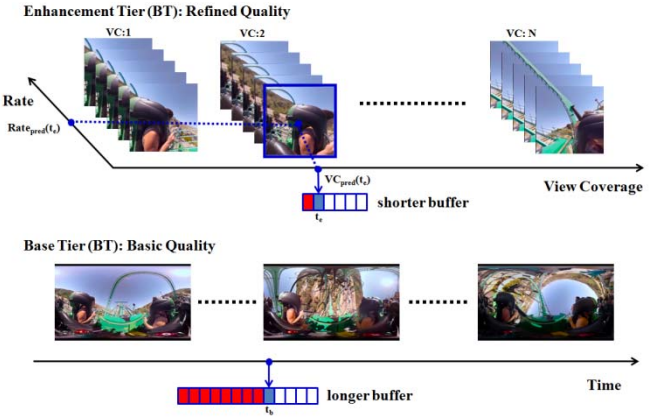

Figure 1. Two-Tier 360 Video Streaming System

Unlike most of the previous solutions, our proposed two-tier scheme simultaneously addresses the accommodation of unexpected head motions and network variations, to provide users with a stable immersive experience. Furthermore, our proposed scheme is source representation independent. Any source projection method (cube-map [4] or equi-rectangular) and source bit-allocation approach (e.g., region-adaptive smoothing [6]) can be easily incorporated.

III. EXPERIMENTS

To evaluate the potential of our proposed framework, we conducted trace-driven simulations. In this section, the detailed setup procedures are presented and two benchmark solutions are simulated for performance comparisons.

A. Video Traces

We downloaded two 360 videos (2160x3840, 30Hz) from Youtube, as shown in Fig. 2. These videos are stored in eqi-rectangular format so that each video can be coded as a conventional 2D video. For the experiments conducted here, we partition the 360 view scope in the azimuthal direction to form different VCs. Each VC spans 120 degree in the azimuthal direction (corresponding to 1280 pixel columns in the equi-rectangular representation). The ET includes a total of 12 overlapping VCs with 30 degree stride (i.e., 320 pixel columns). Each VC is also a rectangular video with a resolution of 2160x1280. We code each original 360 degree video directly using the *x264* encoder [8] at a target rate of R_0. To generate the ET chunks, we calculate the difference between the original 360 video and that decoded from the BT chunks. We encode the difference signal in each VC at two target rates R_1 and R_2 to generate the ET chunks. Each video is encoded with a Group of Picture (GOP) length of 1 second so that the generated chunks of 1-second video are independently decodable. Please note that the actual number of bits for each chunk may fluctuate slightly around the specified target bitrate due to the imperfect rate control. A sample video trace is provided in Fig. 3.

Figure 2. Sample Frames in Test Videos ("Roller Coaster" and "Tarzan")
(Video 1 source: https://www.youtube.com/watch?v=-xNN-bJQ4vI)
(Video 2 source: https://www.youtube.com/watch?v=Hnwd9jMFn8s)

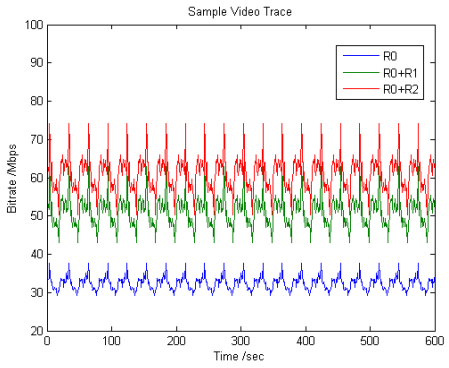

Figure 3. Sample Video Rate Trace for "Roller Coaster" at target rates of R_0= 30 Mbps, R_1= 17 Mbps and R_2 = 47 Mbps. The trace is generated by looping the actual rate trace of 30 second till 600 second.

B. Network Traces

To evaluate the performance of the proposed system in dynamic networks with significant BW variation, we use the traces collected over a 3.5G HSPA cellular network using the methodologies described in [9]. A sample trace is provided in Fig. 4. These traces represent the most typical bandwidth variations in a cellular network. We scale up the original BW values to accommodate the 4K/30Hz 360 degree video bitrate range (i.e., up to 150 Mbps).

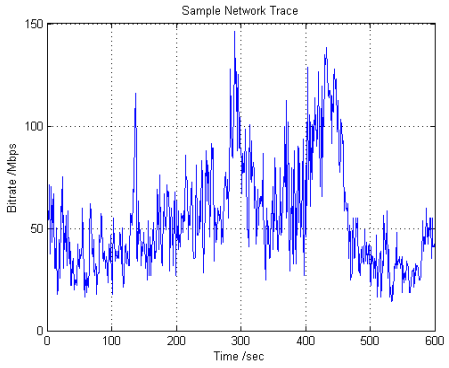

Figure 4. Sample Network Trace after amplitude scaling

C. View Direction Traces

We collected the view direction traces from 4 users as follows. Each user wears a Google Cardboard [10] with a Motorola Nexus-6 smart-phone playing our test 360 videos. Simultaneously, a head tracker [11] equipped on Cardboard dynamically transmits motion data (e.g., yaw, pitch, roll, etc.) to a nearby PC for data recording. For simplicity, we only focus on the horizontal head motion (i.e., yaw) ranging between 0 degree and 360 degree. In Fig. 5, a sample trace generated by concatenating the view direction data from 4 users over the "Roller Coaster" video is provided.

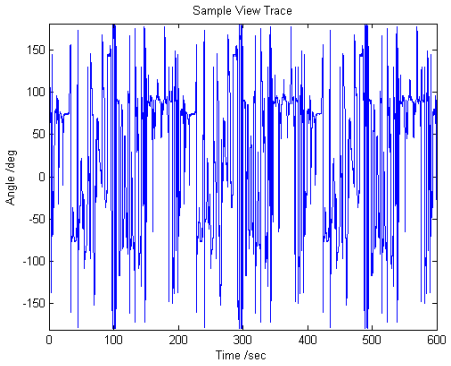

Figure 5. A Sample Viewing Direction Trace for "Roller Coaster". This trace is generated by concatenating the traces from 4 users and looping the combined trace till 600 seconds

D. Proposed "Two-Tier" Solution (TTS)

For simplicity, we consider the following simple two-tier setup for proof-of-concept: the BT stores the 360 degree video segments (each with 1 second of video) coded at a fixed low bitrate R_0. The ET stores all possible VCs (120 degree span for each with 30 degree stride). Each VC segment from ET is coded at two different rates (R_1 or R_2). The initial buffer length for BT is set to 20 seconds. Upon complete reception of a chunk, the client estimates the sustainable transmission rate for the next chunk to be equal to the average throughput for downloading the last received chunk (from either BT or ET), i.e., $\widehat{R}_{n+1} = B_n/T_n$, where B_n is the size of the last received chunk and T_n is its transmission time. Assuming that the last chunk is received at time t, which is between the display time of Segment $n-1$ and Segment n, the client will consider whether it can request an ET chunk for Segment $n+1$. It predicts the view direction for Segment $n+1$ using the current view direction, i.e., $\widehat{V}_{n+1} = V_n$ and determines a VC centered at $\hat{\theta}_{n+1}$. Let $R_{n+1,1}$ and $R_{n+1,2}$ denote the actual rate of the low-quality chunk and the high-quality chunk for this VC, respectively. If the predicted bandwidth suggests the high-rate chunk can be completely received by its display time $n+1$ (namely, $\widehat{R} \cdot (n+1-t) \geq R_{n+1,2}$), then the client will request the high-rate chunk. Otherwise, if $\widehat{R} \cdot (n+1-t) \geq R_{n+1,1}$, the client will request the low-rate ET chunk. Otherwise, the user will request the next BT chunk that is not yet in the BT buffer, which is usually for a future Segment $n+k$, with $k > 1$. We select the target video rates based on the network trace cumulative distribution function (CDF). Specifically, $\widehat{R}_0$ (30 Mbps), $\widehat{R}_1 = R_0 + R_1$ (47 Mbps) and $\widehat{R}_2 = R_0 + R_2$ (77 Mbps), are chosen at 10%, 40% and 80% percentile of the network bandwidth CDF, respectively.

At time n, the receiver will decode and render the video for Segment n according to the view direction trace. If a VC chunk has been received in the ET buffer, it will be decoded and used to enhance the BT decoded 360 view in that VC. Otherwise the 360 video decoded from BT will be directly used to render the desired views during this segment. Note that it is possible that the viewer may continuously change the viewing directions during this period and hence we always resort to the reconstructed 360 view over this period.

For simplicity, we use the effective video rate (*EVR*) to evaluate the received video quality for the desired view direction for any video frame f in Segment n. Let $R_{n,0}$ indicate the rate of the BT chunk for this segment, and $R_{n,x}$ indicate the rates of the ET chunk for a particular VC, where x is the video rate indicator and can be 1 or 2. We define w_f as the overlapping portion of the desired FOV (based on the view trace) at frame f and the VC of the downloaded ET chunk, w_b the overlapping portion of the desired FOV and the 360 view decoded from the BT. The *EVR* for frame f is defined as $R(TTS) = w_b R_{n,0} + w_f R_{n,x}$. Here we assume that the rendered view at any frame covers 120 degree in azimuthal direction so $w_b = 1/3$. In the rare case when BT chunk is unavailable for this segment because the 360 video buffer is empty at the display time, then *EVR* = 0.

E. Benchmark Solution 1 (BS1) - "All 360"

BS1 simulates the typical DASH streaming framework for 360 degree videos, in which videos in the entire 360 view range are pre-encoded using multiple rates. For a fair comparison with TTS, we assign 3 different rate versions (i.e., $R_L = R_0$, $R_M = R_0 + R_1$, $R_H = R_0 + R_2$). Each video chunk contains 1-second video. The client display buffer is pre-loaded with 20-second video. For simplicity, the client estimates the sustainable transmission rate for the incoming segment $n+1$ to be equal to the measured throughput for downloading the last received chunk and accordingly chooses a rate version to request over the next segment. The *EVR* for the desired view over each segment is the rate of the downloaded chunk scaled by a factor w_b and is 0 when the buffer is empty. Recall that $w_b = 1/3$ in our setting.

F. Benchmark Solution 2 (BS2) - "FOV Streaming"

BS2 stores only pre-coded VCs (each VC has 120 degree span, and 30 degree stride between two adjacent VCs) with the same settings as our ET but coded directly with three possible rates ($R_L = R_0$, $R_M = R_0 + R_1$, $R_H = R_0 + R_2$). At time n, it predicts the view direction at $n+1$ using the current view direction, namely, $\hat{\theta}_{n+1} = \theta_n$ and requests the VC centered at $\hat{\theta}_{n+1}$. If the requested segment does not arrive completely within 1 second, then we set *EVR* = 0 over that second. Otherwise, we use the portion of the downloaded VC that overlaps with the desired view to calculate the

number of bits for each frame, i.e., $R(BS2) = w_f \cdot R$, where $R \in \{R_L, R_M, R_H\}$.

IV. SIMULATION RESULTS

Our proposed TTS is simulated and compared with the two benchmarks. We evaluate the system performance using the decodable *EVR*, as defined in Sec.III.D, Sec.III.E, and Sec.III.F. Two test videos (each of 1 to 2 minutes long and played in loops) are simulated over 4 different network traces (each of 600 seconds). Table I summarizes the *EVR* using different BW traces for the three solutions.

The sample video and view direction traces are played in loops for a duration of 10 minutes over different network traces. To illustrate, the sample results using network trace 3 (as plotted in Fig. 4) are provided in Fig. 6.

In BS1, the long buffer setting effectively smooths the network variation and provides a stable display. However, the effective rate is only one third of the encoded video rate covering the 360 view span.

In BS2, when BW is large, the high-rate video chunk can be more safely delivered when BW prediction and view prediction are accurate. However, when the BW suddenly changes from high to low, and is not predicted accurately and promptly, the requested high rate chunk may not arrive in time, resulting in annoying video "freeze" and degrade the effective video rate.

In TTS, the advantages of the two benchmark systems are integrated. On one hand, the BT long buffer can effectively smooth the network variations to improve client experience by avoiding video "freeze". On the other hand, the ET chunks received serve as the "quality booster" when there is sufficient bandwidth. As shown in Table I, the gain in *EVR* is 24.7% over BS1 and 30.0% over BS2.

V. CONCLUSIONS

In this paper, a novel two-tier framework is proposed to enhance the bandwidth utilization for 360 video streaming, while effectively handle the network bandwidth and viewing direction dynamics. Through our trace-driven simulations, even with only naive video fetching policies, the proposed system framework can achieve over 25% gain on average in terms of effective video rate through rate-based switching between the base tier chunks and the enhancement tier chunks. For the future research, we will (1) improve the view prediction and bandwidth prediction, (2) explore the video rate request policy using buffer based solutions or Markov decision processes (MDP), (3) extend the current framework to multiple tiers and (4) develop visual quality metrics to evaluate the streamed 360 videos.

Test Videos	Network Trace 1			Network Trace 2			Network Trace 3			Network Trace 4		
	BS1	BS2	TTS	BS1	BS2	TTS	BS1	BS2	TTS	BS1	BS2	TTS
Tarzan	5.80	5.12	**6.65**	12.13	11.97	**15.45**	19.61	19.23	**23.15**	20.15	19.02	**22.40**
Roller Coaster	5.33	5.11	**7.17**	11.37	11.20	**17.09**	19.66	19.49	**22.61**	18.52	17.98	**23.63**

TABLE I AVERAGE RECEIVED VIDEO EFFECTIVE RATE OVER TIME (Unit: Mbps)

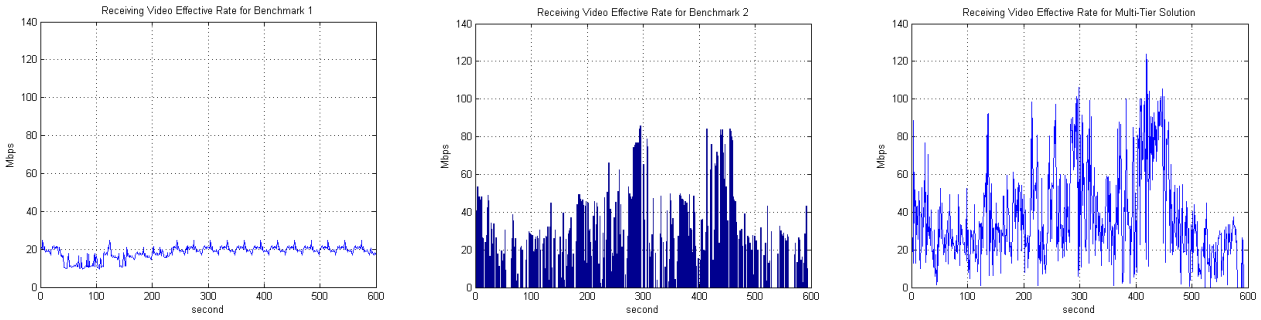

Figure 6. Performance Evaluation using Benchmark 1 (left), Benchmark 2 (middle) and proposed Two-Tier solution (right)

REFERENCE

[1] YouTube, "www.youtube.com" .

[2] Facebook, "www.facebook.com" .

[3] Y.-K. Wang, Hendry, M. Karczewicz, "Tile based VR video encoding and decoding schemes," Doc. JCTVC-X0077, May 2016.

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

[5] Next-generation video encoding techniques for 360 video and VR. {https://goo.gl/DvYivQ.}

[6] M. Budagavi; J. Furton; G. Jin; A. Saxena; J. Wilkinson; A. Dickerson, "360 Degrees Video Coding Using Region Adaptive Smoothing, " Proc. IEEE International Conference on Image Processing (ICIP), pp. 750 - 754, Canada, 2015.

[7] F. Qian, L. Ji, B. Han, V. Gopalakrishnan, "Optimizing 360 Video Delivery Over Cellular Network," Proceedings of the 5th Workshop on All Things Cellular (ATC), New York City, New York, USA, 2016.

[8] x264, "http://www.videolan.org/developers/x264.html".

[9] E. Kurdoglu, Y. Liu, Y. Wang, Y. Shi, C. Gu, and J. Lyu. "Real-time Bandwidth Prediction and Rate Adaptation for Video Calls over Cellular Networks," In Proc. of the ACM Multimedia Systems Conference, Klagenfurt, Austria, 2016.

[10] Google Cardboard. {https://vr.google.com/cardboard/index.html.}

[11] OpenTrack: head tracking software. {https://github.com/opentrack/opentrack.}