Panoramic Video Delivery Based on Laplace Compensation and Sphere-Markov Probability Model

Tongyu Zong[1], Zhengzhong Tu[1], Li Ai[2], Xueliang Xi[2], Yize Jin[1], Xiaoyang Zeng[1], and Yibo Fan[1]*,
Member, IEEE
[1]State Key Lab of ASIC and System, Fudan University, Shanghai, China
[2]China North Industries Group Corporation, Beijing, China
tyzong16@fudan.edu.cn, *fanyibo@fudan.edu.cn

***Abstract*—Virtual Reality (VR) is on the rise nowadays. To save bandwidth and ensure high quality at the same time, only the tiles within field of view (FOV) of the user should be transmitted in high quality while other tiles in low quality or even not transmitted. However, if a user's viewpoint moves fast while watching a panoramic video, low-quality content will always appear in the FOV because of the round-trip time (RTT) delay. Therefore, prediction of viewpoints is used to reduce low-quality area in the FOV. In this paper, we propose methods based on Laplace compensation and Sphere-Markov probability to increase high-quality area in the FOV. Quality value could be increased at most 60.6% and 137% respectively by the two methods.**

I. Introduction

Independent production of VR videos has increased due to the development of omnidirectional cameras, multi-surface projection, network transmission and many other kinds of technology [1]. Due to the prospect of VR, many major players in computer industry have introduced their own headsets such as Google Cardboard and Daydream, HTC VIVE, Sony PlayStation VR and Samsung GearVR [2]. Users can interactively change viewpoints and dynamically view any part of the captured scene they desire [13].

However, transmitting the entire panoramic video over networks with limited bandwidth is challenging. Furthermore, there exists RTT [3] between the time a client starts to send its viewport information to the server and the time it receives the corresponding streams covering the FOV of that viewport, so there may be a discrepancy between the receiving high-quality content and the actually needed content. Therefore, some low-quality contents may be seen within the FOV. Consequently, predictive approaches should be exploited to ensure quality of experience (QoE).

Traditionally, people predict users' viewpoint after a fixed RTT using the model of constant angular velocity or acceleration under the assumption that position, velocity and acceleration could be perfectly obtained [4]. The method performs passably under low RTT delay, but QoE decreases sharply when RTT gets larger.

In this paper, we first provide an effective method based on Laplace compensation to improve QoE under low RTT delay, and then we also propose another way based on Sphere-Markov probability aiming to keep a high QoE under high RTT delay.

We use spherical projection in this paper because it is convenient to calculate predicted results. References [5] and [6] have introduced more kinds of projection which we will adopt in our future research. Furthermore, tiling method is adopted so that only the related tiles will be transmitted to the client, which ensures low bit rate (BR) [7].

In section II, we introduce our panoramic system adopting the tile-reordering method. In section III, we explain some traditional methods. In section IV, we focus on the two predictive approaches proposed by us. In section V, we show the results of our experiments. In the last section, we summarize the conclusions.

II. Tile Reordering

We tile the content into a regular grid to enable random access to regions of interest (ROI) [8]. In this way the server does not have to transmit the entire content, or it will waste too much bandwidth.

The simplest system without reordering the tiles is described as below: first, two streams are encoded from the same sequence in constant bit rate (CBR) mode. They are of different quality levels: one of the streams is of high quality, which means its CBR is large, and another is of low quality. When a user is watching the panoramic video, the client will continually send its viewpoint vector to the server. After half of the RTT, the server will receive the information and immediately transmit to the client high-quality streams of the tiles overlapped with the FOV and low-quality streams of tiles completely outside the FOV. Then after another half of the RTT, the client will receive the streams. It will decode them using Fast Forward mpeg (FFmpeg) and render these received streams using Open Graphics Library (OpenGL) and finally the user could see the content of the stream. Traditionally, HEVC Test Model (HM) could be used, which supports tile encoding mode, to encode a sequence. Fig. 1 shows an example of a frame consisted of 18 tiles. Suppose that red tiles are overlapped with the current FOV and should be transmitted in high quality. Then like Fig. 2 shows, the server will package each of the tiles into a Real-time Transport Protocol (RTP) packet then transmit them one by one in a constant order. The client could decode a frame only when it has received all of the tiles making up the whole frame, which definitely leads to a larger RTT delay.

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18

Fig. 1. Example: a frame consisted of 6x3 tiles. Red tiles are overlapped with the current FOV and should be transmitted in high quality.

This work was supported in part by the National Natural Science Foundation of China under Grant 61674041, in part by the STCSM under Grant 16XD1400300, in part by the State Key Lab of ASIC & System under Grant 2015MS006.

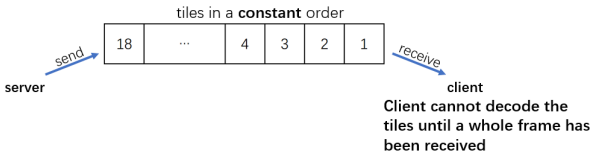

Fig. 2. Server transmits the tiles in a constant order.

Instead of encoding the whole sequence in tile-encoding mode, we treat each tile as a frame, which means we separate an original frame into a number of rectangles of the same size, and each rectangle is a frame to be encoded independently. For convenience, we still call the small rectangular frames "tiles". So, the point is the needed tiles could be first transmitted by the server, others subsequently, which is depicted by Fig. 3.

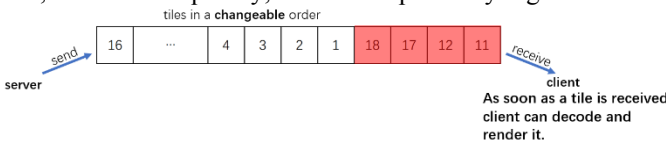

Fig. 3. Tile Reordering: the needed tiles could be first transmitted by the server.

III. Traditional Predictive Method

A. Constant Angular Velocity

If the network delay is small, within the delay we can assume that the angular velocity of the user is constant, both its value and direction. Suppose that the current viewpoint is $\boldsymbol{x_{current}} = (x_{c0}, x_{c1}, x_{c2})$, angular velocity is ω, whose direction is $\mathbf{direc} = (d_0, d_1, d_2)$ and RTT is known. Then we could figure out the angle that the user has swept during RTT:

$$\theta = \omega \cdot \mathrm{RTT} \tag{1}$$

The predicted viewpoint $x_{predicted} = (x_{p0}, x_{p1}, x_{p2})$ could be figured out as below:

$$x_{p0} = x_{c0}cos\theta + x_{c2}sin\theta \cdot d_1 + x_{c1}sin\theta \cdot d_2 \tag{2}$$

$$x_{p1} = x_{c1}cos\theta + x_{c1}sin\theta \cdot d_0 + x_{c0}sin\theta \cdot d_1 \tag{3}$$

$$x_{p2} = x_{c2}cos\theta + x_{c0}sin\theta \cdot d_2 + x_{c2}sin\theta \cdot d_0 \tag{4}$$

If a tile is overlapped with the FOV of the predicted viewpoint, the server will package the tile of the highest quality and send it out. Otherwise, if the tile is located outside the FOV, the lowest quality of this tile will be chosen.

The result of our experiment shows that this simple model could provide obviously better QoE than the nonoptimized system even when RTT is very small. And when RTT gets larger, the improvement will become more obvious. However, it also shows that when RTT is large, QoE based on this model cannot make users satisfied.

B. Constant Angular Acceleration

The only difference between the model in this part and the one introduced above is that angular acceleration a is considered, whose direction vector is assumed in the same straight line with ω. Therefore, the sweeping angle during the RTT should be changed as below:

$$\theta = \omega \cdot \mathrm{RTT} + 0.5a \cdot RTT^2 \tag{5}$$

The formula of $x_{predicted}$ is the same as (2) to (4).

IV. Proposed Methods

A. Laplace Compensation

Although the trajectory of users' movement could be predicted in some way, any prediction causes dynamical error and side effects [9], so prediction is always not accurate enough. At any instant time, a user could move at any velocity, any acceleration and toward any direction. And QoE decreases dramatically when RTT gets a little larger. Therefore, it is not tenable enough to only deal with one predicted viewpoint, instead, we should deal with a region.

Fig. 4(a) depicts that the current viewpoint is $\boldsymbol{x_{current}}$, the instant angular velocity is $\boldsymbol{\omega}$ and acceleration is $\boldsymbol{a}$. The red arrow indicates the direction of velocity and acceleration. It is tenable that the direction of acceleration is either the same as velocity or opposite to it when RTT is small. And in Fig. 4 they are in the same direction. Fig. 4(b) depicts the predicted viewpoint as well as the actual viewpoint after one RTT delay. They are almost the same especially when RTT is small, but in most cases, predicted viewpoints deviate more or less from actual ones.

The bright red tiles are of the highest quality according to $\mathbf{x_{predicted}}$, which is obtained using (1) to (4). Traditional approaches only transmit these tiles. Although they cover most area of the actual FOV, there is always an error field not being covered, which leads to bad QoE. We also notice that the error field always appears along the direction of acceleration, so tiles in the extending path should also be transmitted in an acceptable quality. That's exactly what compensation means.

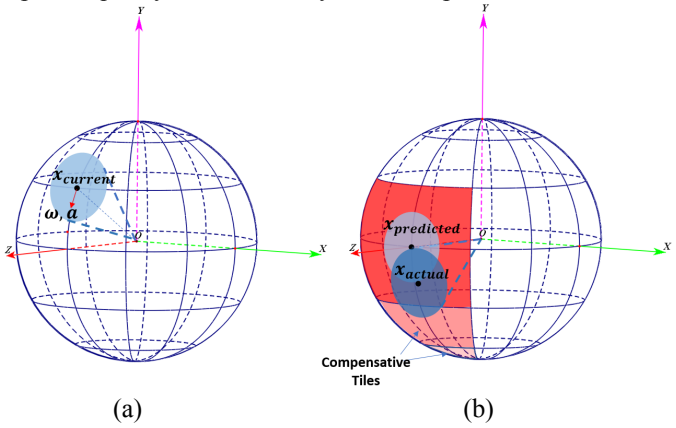

Fig. 4. (a) depicts current viewpoint $\boldsymbol{x_{current}}$ at time t. The direction of acceleration is **the same as** angular velocity. The light blue circular area is FOV of the current viewpoint. (b) depicts the predicted viewpoint and the actual viewpoint at time $t + \mathrm{RTT}$. The bright red tiles are of the highest quality according to $\mathbf{x_{predicted}}$, and the light red and lighter red region are **compensative tiles**. The lighter the color is, the lower quality the tiles are in.

The tiles overlapped with the extending path are called compensative tiles. Obviously quality level of a compensative tile is related to the extending angular distance d between the predicted viewpoint and this tile. We need to emphasize that the distance must be the projection distance along the direction of the acceleration, which is depicted in Fig. 4(b). The quality is related to acceleration, too. The larger acceleration the viewer has, the higher quality compensative tiles should be in.

Therefore, we exploit Laplace distribution:

$$quality\ level_{\text{tile}} = exp(-|d_{tile}|/b) \quad (6)$$

$$\text{tile} \in \{Compensative\ Tiles\} \quad (7)$$

$$b = a_0 + k \cdot |a| \quad (8)$$

$quality\ level_{tile}$ indicates a tile's quality level, which lies in $[0,1]$. Due to the fact that we adopt CBR encoding mode, quality of a tile could be represented by its encoding BR. And a maximal value BR_{max} will be designated for each sequence. Accordingly, BR of a tile can be written as:

$$BR_{tile} = quality\ level_{tile} \cdot BR_{max} \quad (9)$$

d_{tile} in (6) is the projection distance between a compensative tile and the predicted FOV. b is called scale parameter. The larger the scale parameter, the more spread out the distribution [10], in other words, the higher quality compensative tiles are of. Consequently, b should be positively correlated with acceleration a. We simply let b be in proportion to a, which works well enough in our experiment. Note that if acceleration or velocity equals to 0, there will not be any compensation. We have also adopted Gaussian distribution, but it performs worse. Furthermore, we should define a compensative range between the predicted FOV and compensation boundary. The larger RTT and acceleration are, the larger the range is.

$$range = 0.5 \cdot a \cdot RTT^2 \quad (10)$$

In another case, the direction of acceleration is opposite to that of velocity. Except compensative tiles lie in the opposite direction, the compensative method is all the same.

B. Sphere-Markov Probability Model

The second proposed model for prediction is called Sphere-Markov probability model. Firstly, "Sphere" means the model is based on a spherical projection space and we use "Markov" because the process that a viewer switches perspective can be supposed to be a Markov Process. There are works utilizing Markov process in mobility prediction such as [11], but till now no work uses this model to predict viewpoints in spherical panoramic videos. The motivation of this approach is when RTT gets larger, modeling trajectory does not work well, which is shown in our experimental results. Instead, relying on prior probability is a better solution.

For each viewpoint, we need to know the probability distribution of the next viewpoint a time interval (Δ) later. We should first define finite discrete viewpoints which approximate actually infinite viewpoints on the spherical surface. We divide the entire spherical surface according to longitude and latitude. Rotation angle between neighboring longitudes is u degrees and azimuth angle between neighboring latitudes is also u degrees. We regard those intersection points of latitudes and longitudes as well as the north pole and south pole, totally n points, as viewpoints:

$$n = 360 \cdot u^{-1} \times (180 \cdot u^{-1} - 1) + 2 \quad (11)$$

In the experiment, we set $\mathrm{u} = 15^{\circ}$. Then we let 100 people to watch our panoramic videos and recorded viewpoints every Δ ms. Finally, we obtained the Markov probability matrix $\mathrm{P} \in R^{n \times n}$.

$$\mathrm{P} = (\boldsymbol{p_1}, \boldsymbol{p_2}, \dots, \boldsymbol{p_n})^T \quad (12)$$

If an observer's viewpoint at time t is $\mathrm{x}[t] = i$, then the probability of his or her looking at $\mathrm{x}[t + \Delta] = j$ (at time $t + \Delta$) is $P_{i,j}$. We use matrix Q to represent probability distribution after one RTT delay:

$$\mathrm{Q} = P^{RTT/\Delta} \quad (13)$$

$$\mathrm{Q} = (\boldsymbol{q_1}, \boldsymbol{q_2}, \dots, \boldsymbol{q_n})^T \quad (14)$$

$\boldsymbol{q_i}$ is exactly the probability distribution of the after-delay viewpoint of this observer. For any after-delay viewpoint $j \in \{1,2, \dots, \mathrm{n}\}$, quality level of the tiles which are overlapped with FOV of j should be in proportion to $Q_{i,j}$. If a tile is overlapped with both j's and k's FOVs, its quality level should be decided by the larger one between $Q_{i,j}$ and $Q_{i,k}$. If $Q_{i,j}$ is the largest, the quality level of this tile at time $t + \Delta$ will be:

$$quality\ level_{tile} = \frac{Q_{i,j}}{the\ maximal\ element\ in\ \boldsymbol{q_i}} \quad (15)$$

C. Evaluation of QoE

In the experiment, for each sequence, every 33ms we calculate Weighted Area Ratio (WAR) of FOV.

$$\mathrm{WAR}_t = \frac{\sum quality level_{tile} \times S_{tile}}{S_{FOV}} \quad (16)$$

Subscript t means current time. S_{tile} is a tile's area within the current-time FOV, and the area is weighted by the tile's quality level according to the particular algorithm mentioned above. The quality level is between 0 and 1. All the tiles' weighted area should be summed up and then divided by area of the whole FOV. For each sequence, we could obtain a sequence of WARs and then their average value is what we call quality value:

$$\text{quality value} = \frac{\sum_t WAR_t}{the\ number\ of\ WAR_t} \quad (17)$$

V. Experiments And Results

A. Experimental Setup

The sequences we have used for testing our models could be obtained from [12]. They are prepared for spherical projection and resolution of most of them is 3840x1920. Few of them need to be interpolated to this resolution during the experiment.

Firstly, we used HM to encode each sequence into 10 different quality levels in CBR mode. We set the maximal bit rate per tile up to 280kbps and each frame is consisted of 72 tiles, which are of the same size: 320x320. Therefore, the maximal bit rate of the whole sequence is $280 \times 72 = 20160$ kbps. The n^{th} quality level of a tile is $(280 - 28 \times (\mathrm{n} - 1))$kbps, in which $\mathrm{n} = \{1,2, \dots, 10\}$. We set frame rate (FR) to 30fps. In the experiment, the server will transmit streams of tiles, whose quality level will be decided according to the models introduced above. The circular FOV of the client is set to 90 degrees.

We monitored the moving trails of 100 users while they were watching the video. Every 33ms the client sends its information to the server, then after a RTT delay, the client would receive corresponding streams and also arrives at a new viewpoint. Then we could calculate the WAR of its FOV. Finally, we average all the WARs to obtain quality value. We set RTT to different values from 33ms to around 1000ms and compare the quality value and bit rate of each method.

B. Experimental Results

Fig. 6 demonstrates that for the sequence *rollercoaster*, Laplace compensation is always the best way when RTT is below 430ms. If RTT is larger than 430ms, Sphere-Markov model should be exploited. We use $RTT_{threshold} = 430\text{ms}$ to represent the intersection point of the two models. The server should adopt this strategy: RTT between the server and a client should be measured first. If RTT is below $RTT_{threshold}$, the server should take the model of Laplace compensation, and quality value can be increased at least 3.33%, at most 60.6% compared to the model without prediction; otherwise, the server ought to use Sphere-Markov probability model, and the quality value could be increased at least 49.8%, at most 137% compared to the model without prediction. Of course, different sequences have different $RTT_{threshold}$. Furthermore, for the sequence *rollercoaster*, the model of constant angular velocity increases the quality value only 37.7% at most and the model of constant acceleration always performs even worse than it.

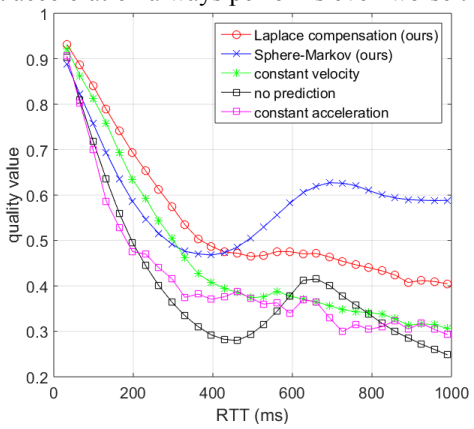

Fig. 5. Sequence ***rollercoaster***. Quality values of the models of Laplace Compensation, Sphere-Markov Probability, constant velocity, no prediction, constant acceleration.

Fig. 7 shows the bit rate performance. As mentioned in Part A of this section, transmitting the whole frame consumes a bandwidth of 20Mbps, which will cause too much burden. From Fig. 7 we can see bit rate of the Laplace-compensating model is at least 4.3Mbps (save 78.7%) and converges to around 5.6Mbps (save 72.2%); bit rate of Sphere-Markov probability model is nearly in proportion to RTT, so if we want high-quality QoE when RTT is large, more bandwidth must be consumed.

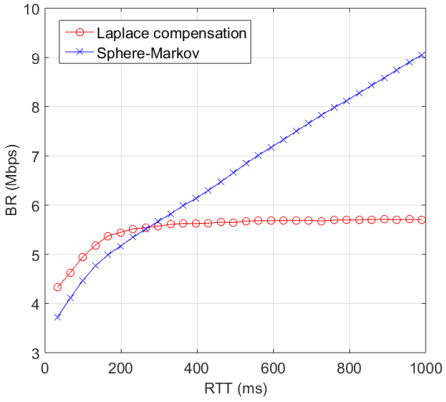

Fig. 6. Sequence ***rollercoaster***. Bit rate of the models of Laplace Compensation, Sphere-Markov Probability.

Results of other sequences are similar to *rollercoaster*. Some of them could be seen in [14].

VI. Discussions And Conclusions

In this paper, we first introduce our server-client system based on tile reordering. We have also described traditional methods aiming to model the motion of viewers. Then we propose a Laplacian compensative method as well as an approach based on Sphere-Markov probability to improve QoE. To achieve the best QoE, we should first obtain $RTT_{threshold}$ of each sequence from enough records of many viewers' watching viewpoints. Then, RTT between the server and a particular client should be measured. If RTT is below $RTT_{threshold}$, the server should adopt Laplacian Compensative method to decide quality level of each tile at each instant time; otherwise the Sphere-Markov Probability model.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant 61674041, in part by the STCSM under Grant 16XD1400300, in part by the State Key Lab of ASIC & System under Grant 2015MS006.

Reference

[1] Virtual reality. [Online]: https://en.wikipedia.org/wiki/Virtual_reality
[2] El-Ganainy T, Hefeeda M, "Streaming virtual reality content," arXiv preprint arXiv:1612.08350, 2016.
[3] Lungaro P, Tollmar K, "QoE design tradeoffs for foveated content provision," in Quality of Multimedia Experience (QoMEX), 2017 Ninth International Conference on. IEEE, 2017, pp. 1-3.
[4] Azuma R, Bishop G, "A frequency-domain analysis of head-motion prediction," in Proceedings of the 22nd annual conference on Computer graphics and interactive techniques. ACM, 1995, pp. 401-408.
[5] Yu M, Lakshman H, Girod B, "Content adaptive representations of omnidirectional videos for cinematic virtual reality," in Proceedings of the 3rd International Workshop on Immersive Media Experiences. ACM, 2015, pp. 1-6.
[6] Li J, Wen Z, Li S, Zhao Y, Guo B, Wen J, "Novel tile segmentation scheme for omnidirectional video," Image Processing (ICIP), in 2016 IEEE International Conference on. IEEE, 2016, pp. 370-374.
[7] Gaddam V R, Riegler M, Eg R, Griwodz C, Halvorsen P, "Tiling in interactive panoramic video: Approaches and evaluation," IEEE Transactions on Multimedia, vol. 18, no. 9, 2016, pp. 1819-1831.
[8] Rondao Alface P, Macq J F, Verzijp N, "Interactive Omnidirectional Video Delivery: A Bandwidth□Effective Approach," Bell Labs Technical Journal, vol. 16, no. 4, 2012, pp. 135-147.
[9] Kijima R, Miyajima K, "Measurement of Head Mounted Display's latency in rotation and side effect caused by lag compensation by simultaneous observation—An example result using Oculus Rift DK2," in Virtual Reality (VR). IEEE, 2016, pp. 203-204.
[10] Scale parameter. [Online]: https://en.wikipedia.org/wiki/Scale_parameter
[11] Chan A, Li F W B, "Utilizing massive spatiotemporal samples for efficient and accurate trajectory prediction," IEEE Transactions on Mobile Computing, vol. 12, no. 12, 2013, pp. 2346-2359.
[12] Sequences. [Online]: http://soc.fudan.edu.cn/demo/sequence/
[13] Liu T M, Ju C C, Huang Y H, Chang T S, Yang K M, Lin Y T, "A 360-degree 4K× 2K pan oramic video processing Over Smart-phones," in Consumer Electronics (ICCE), 2017 IEEE International Conference on. IEEE, 2017, pp. 247-249.
[14] Some results. [Online]: http://soc.fudan.edu.cn/vip/attachments/download/6005/results%20of%20paper.pdf