SDN-Assisted Adaptive Streaming Framework for Tile-Based Immersive Content Using MPEG-DASH

Shuai Zhao, Deep Medhi
Department of Computer Science & Electrical Engineering
University of Missouri–Kansas City, USA
{Shuai.Zhao, DMedhi}@umkc.edu

***Abstract*—Video streaming over the internet for new 3D immersive media such as Virtual Reality and 360-degree videos are drawing great attentions from both consumers and researchers in recent years. One of the biggest challenges in streaming such 3D media is the high bandwidth demands. While traditional 2D video streaming is still dominating network peak traffic, new inventions are accelerating the adoptions of immersive contents and devices. A new Tile-based video is introduced in both video codec and streaming layer to reduce the transferred media size. Dynamic adaptive streaming over HTTP is becoming one of the de facto effective adaptive streaming approaches that can fully utilize the existing physical IP network infrastructure. In this paper, we propose a tile-based streaming framework using software-defined networking. By prioritizing streaming flows based on the region of interests, our approach can improve user's quality of experience (QoE).**

***Index Terms*—MPEG-DASH, Spatial Relationship Description; Software-Defined Networking; DASH Streaming; Immersive VR/360**

I. Introduction

The public interest of immersive devices such as head-mounted displays (HMD) for Virtual Reality (VR), 360-degree video cameras for capturing immersive contents and 3D playback support from the commercial website such as YouTube are drawing great attentions from both consumers and researchers. Compare with regular 2D flat video contents, the 360 VR videos are extremely bandwidth intensive especially with the 4K/8K video resolution being widely accepted as a functional minimum resolution for current HMDs, while 8K or higher is desired. Therefore, a major challenge is how to efficiently transmit these bulky 360 VR videos to bandwidth-constrained wireless VR HMDs at acceptable quality levels given their high bitrate requirements. Recommended VR/360-degree video display aspect ratio from YouTube is 2:1 for monoscopic or panoramic videos and 1:1 for stereoscopic videos with a ratio of 2:1 per eye for better user QoE. The file size is double w.r.t the original 2D videos. Figure 1 depicts how video content size can grow with various resolution.

A recently proposed tile-based approach is attempting to reduce the transfer size based on user's region of interests (ROIs). In a typical scenario for streaming VR/360 videos, the user usually views only a portion of the video at a time, called field-of-view (FoV). As a result, there is a huge waste of bandwidth for streaming content to the client that is not visible to the user. By knowing the user ROIs, we can stream it with high quality while minimizing the quality of the rest of the video and thereby, saving the user bandwidth. In a video tiling scenario, the video is partitioned to multiple tiles and depending on the users viewable area, we stream the overlapping tiles. Tiling has proven useful in domains such as online video lectures and sports.

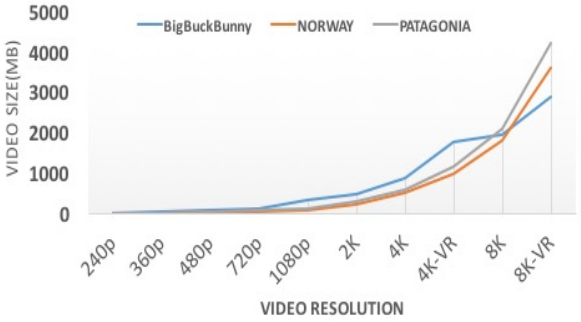


Fig. 1. Video Content Size Comparison

Dynamic adaptive streaming over HTTP (MPEG-DASH) [1] was initially designed for efficient streaming of 2D flat videos. It has become one of the de facto effective adaptive streaming approaches that can fully utilize the existing physical IP network infrastructure. However, the current IP network has the limitation of effective dynamic bandwidth allocation, which can lead to suboptimal streaming experience for immersive content consumers.

Software-defined Networking (SDN) is a relatively recent networking paradigm that has been conceived to address certain limiting of IP networking. With decoupled control and forwarding layers, SDN can dynamically optimize network flows traffic based on global network traffic information. It also has a finer quality of service (QoS) control based on assigned flow priorities. Our work aims to investigate how the future immersive video streaming scheme can exploit SDN for better QoE. Specifically, we propose an SDN-based approach to design a tile-based VR/360 streaming platform using DASH. Briefly, we make the following contributions:

1) We present the difference in immersive content stream-

ing between traditional- and SDN-based network.
2) We introduce an SDN-based framework to assist tile-based immersive content streaming.
3) We quantify the benefits brought by our SDN approach using network simulation. The results indicate that our scheme can increase of user's quality of experience.

The remaining of this paper is as follows. Relate work is discussed in Section II. Tile-based streaming support for MPEG-DASH is explained in Section III. We then give a brief overview of background regarding SDN and proposed a framework in Section IV. Section V describes our test environment and experimental results. We conclude the paper in Section VI.

II. Related Work

With modern capturing systems for 4K or Ultra High Definition (UHD) and 8K or ultra high definition television (UHDTV) video, new types of media experiences are possible where end users have the possibility to choose their viewing direction. Also with increased interest of immersive HMD devices for VR/360 content playback, user's viewing experience suffers because of the present limitations in both of existing network framework and video delivery methods.

A great deal of work has been done on both video encoder and delivery methods to reduce transferred media size. Tiling, in the video codec level such as the H.265/HEVC [2], [3] refers to a spatial partitioning of a video where tiles correspond to independently decodable video streams, which takes a divide-conquer approach to encoding and can reduce video content size by half. However, it is currently not widely adopted compared with the H.264/AVC encoding method.

With a 2D flat video, a tiled video can be obtained from a single video by partitioning each frame into multiple frames of smaller resolution and by aggregating the smaller frames coming from the same partition/region of the input frame into a new video. Here, tiles are defined as a spatial segmentation of the video content into a regular grid of individual videos. Similar ideas are also being used for creating tile-based VR/360 contents [4]–[9].

MPEG-DASH is one of de facto effective adaptive streaming approaches that can fully utilize the existing physical IP network infrastructure. It supports the tiling scheme in the Media Presentation Description (MPD) file [10], [11]. By specifying spatial relationship description (SRD) in the DASH MPD file, the client can fetch video segments based on current ROIs such as applications in [12]–[15].New streaming architectures is proposed in [16] to provide an efficient streaming experience. However, such experiments have been conducted using the existing IP network architecture.

Related research such as improving the quality of experience for streaming tile-based videos are also being investigated in [3], [5], [17], [18] using available dataset in [19]. Our work focus on exploring new network traffic engineering using SDN. Our work is the first to exploit how SDN can be used in such immersive streaming scenarios. With new MPEG-I (ISO/IEC 23090) standard on the way and 5G networks' advancing, our quantified testing result shows that SDN can be quite beneficial to boost the development of such technologies.

III. MPEG-DASH Tile-based Streaming Support

A. Daynmaic Adaptive Streaming Over HTTP (DASH)

The main concept behind MPEG-DASH (ISO/IEC 23009-1) is shown in Fig. 2 [20]. A Media Presentation Description (MPD) file is stored on the server and is fed to the client player at the start of video viewing. The MPD file describes how a video is segmented with different video and audio adaptation sets and depicts the metadata of video segments such as segment durations, video/audio codec, bitrate, video resolutions, and how segments are stored indicated by segment reference schemes.

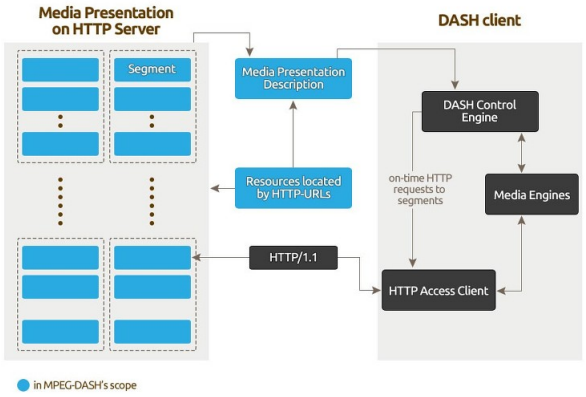


Fig. 2. MPEG-DASH system overview [20]

On the client side, the media player first fetches the MPD file to learn the URLs of all video segments. Thus, the available video segments from a video server can be fetched through HTTP/GET requests.

B. MPEG-DASH Spatial Relationship Description

To make a tiled video, one can resort to either multiple source camera setup, or to partition a single video into multiple frames of smaller resolution. Here, tiles are defined as a spatial segmentation of the video content into a regular grid of independent videos. In addition to supporting segmented video streaming, the DASH standard also allows associating non-timed related information to MPD elements. One of such syntax calls Spatial Relationship Description (SRD) to represent spatial relation for various parts of the same scene. In SRD, the relationship is represented using a scheme URI (@schemeIdUri attribute) and a value (@value attribute). In the value field, one can specify how video tiles are spatially allocated. Figure 3 depicts a tile space for value = $\langle 0, 0, 0, 1, 1, 3, 3 \rangle$. This is a 3×3 tiled coordination with tile's height and length equaling 1. Starting from the lower left corner, tiles' coordinates are numbered.

C. Region of Interests (ROIs) vs Video Tiles

When viewing immersive contents, the ROIs are the current viewport or the viewer's focus. For efficient streaming experience, ideally, one ROI is covered by just one or less than

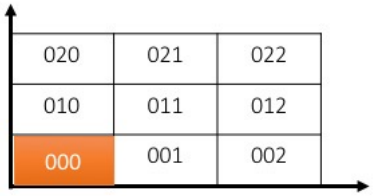


Fig. 3. DASH SRD coordinate example (3 × 3 tiles)

one video tile, which can reduce data size by streaming lower bitrate for non-ROIs tiles and increase, otherwise. However, in the real world use case, a user's ROI falls more like Figure 4, which shows a possible ROI locations for a 3×3 tiled scheme. In such scenarios, it is better for tiles with the number in [010, 020, 011, 021] to get higher bitrates to increase the view quality while the rest of the tiles can be transmitted with lower bitrates. Simply by doing such assessments, the transferred data size can be cut by half with increased QoE significantly.

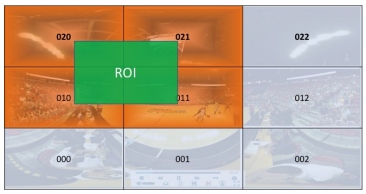


Fig. 4. Tile-based video playback example [8]

On the other hand, the current immersive contents are streamed either by downloading the whole content into player's devices or without any optimization with the user's ROIs. Instead, significant bandwidths are wasted while transferring high bitrate segments that is not necessary. Also, the current IP network does not fully support multipath traffic loading balancing and cannot dynamically adapt network flows based on real-time network information, which can lead to sub-optimal user's quality of experience and waste of bandwidth. Especially, with the humongous immersive media's creation, new approach are investigated in the next section to explore how SDN can be utilized in streaming immersive media.

IV. SDN for Tile-Based DASH Streaming

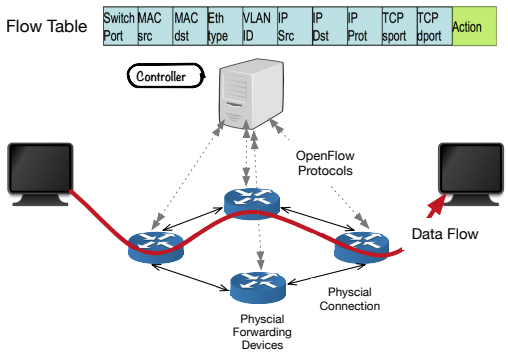


Fig. 5. Software-Defined Network Architecture

A. Background Overview on Software-Defined Networking

Software-Defined Networking provides a dynamic, manageable and cost-effective platform for making it an important platform for high-bandwidth, dynamic nature of today's network applications. Fig. 5 shows the SDN architecture. It decouples the control and data forwarding layers and provides the programming interface for the underlying forwarding devices as well as the upper application layer. The SouthBound and NorthBound APIs are provided as communication channels between the SDN layers. A typical SDN architecture includes two main components: an SDN controller and a traffic forwarding protocol using the forwarding devices. An SDN controller is a software application that manages application flows to enable dynamic and controllable networking environment. SouthBound communication between SDN controllers and forwarding devices can be accomplished using OpenFlow [21] that allows servers to instruct forwarding devices where to send packets.

B. SDN-Assisted Adaptive Streaming Framework for Tile-Based Contents Using DASH

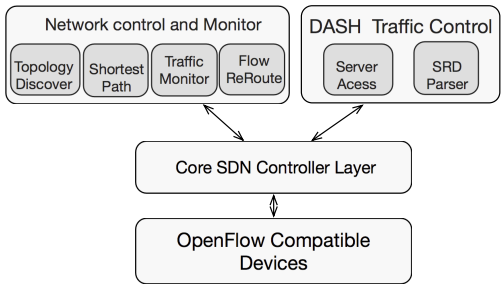


Fig. 6. SDN DASH Experimental Architecture

In this section, we present our proposed SDN-assisted platform design and implementation. Figure 6 depicts the abstract function design for each SDN layers, which includes the SDN control and Application layer.

1) SDN Controller Layer: In the control layer, essential network function is implemented for both physical layer packet forwarding and network layer module in Open System Interconnection (OSI ISO/IEC 7498-1) model. In the packet forwarding module, we apply the network primary forwarding functions including the link layer discovery protocol (LLDP) in the physical network layer. The implementation is based on OpenFlow-compatible forwarding devices. In the network layer, we implemented the forwarding function for the Internet Control Message Protocol (ICMP) messages, which is the key mechanism used to give feedback on network problems that could prevent packet delivery.

Due to the flexibility provided by the SDN framework, we also addressed a new physical layer flooding avoidance mechanism such as for the address resolution protocol (ARP). In a traditional IP network, variations of spanning tree protocols (STP) are widely used to build a loop-free topology. The configuration of such an STP protocol can be cumbersome

and complicated based on the used forwarding devices. We designed and implemented an ARP resolver algorithm that offers smooth ARP package flooding, instead of using a costly STP protocol as would be the case in a traditional IP network environment. It also takes care of ARP cache expiration issues by avoiding to send additional ICMP messages to get an updated ARP entry.

Above the network transport layer, we implemented TCP and UDP packet forwarding functions for application-aware networking. Based on the application layer's port number and protocol type, it will forward packets accordingly. In the traffic monitor module, we implemented lightweight REST-API services to proactively fetch global network information such as port traffic for each forwarding device, flow installation/modification, and traffic details in a managed time interval. The REST-APIs are designed to be lightweight without introducing extra overhead for the SDN controller. One Apache web server collects the pulled results from the REST-APIs and aggregates traffic details to provide any traffic alerts and Traffic Engineering (TE) recommendations.

2) Application layer: In the network control and monitoring layer, the global network topology is discovered where we take an adaptive traffic engineering approach by feeding into a shortest path algorithm module to calculate a path for each pair of network node/hosts on an on-demand basis. The traffic monitor component using REST-APIs' services deployed at the core SDN controller layer proactively pulls network traffic information from the network. If there is any pre-defined traffic priority violation, a traffic reroutes using flow reroute component might happen. From the beginning, for ARP messages for a network request such as Ping, SSH, or other applications, it first looks at the flow table and passes the traffic if there is a current matching flow or checks if it is a ARP broadcasting message otherwise.

Port number based application recognition feature is implemented (such as port 80 is by default for the Apache Web Service). In our controlled network, the port number can be managed/changed via a separate configuration file that is read by our SDN controller. With regards to DASH streaming applications, web server and DASH client control components are implemented to instruct how to install flows over forwarding devices. The modularization of various components provided by different SDN controllers helps the network administrator to control them individually in a manageable way.

The SRD parser module can parse the MPD file and extract the tile coordination based on user's ROI change. Shortest path module can calculate in real-time for all shortest paths between each pair of nodes based on existing network conditions and topology changes. In our test scenarios, higher priority tiles are rerouted using flow reroute module to the non-bottle-necked path to achieve better bandwidth utilization.

V. Experiment Enviroment and Performance Evaluation

In this section, we conduct our test environment setup and show our result based on various number of ROIs. For comparison, we first conduct bitrate tracing using a traditional network and then switch to our proposed SDN network. By showing the increased bitrate and buffer size for user's ROI, it shows the benefits of our proposed framework.

A. Network Topology

Mininet is a well known SDN emulator [22]. The topology is as shown in Fig.6 that is setup with 10 Mbps link bandwidth and 2 ms delay. We set up a network topology as illustrated in Fig. 7. We use Openvswitch (OVS v2.3.1) [23] as our forwarding devices. One DASH Apache web server and DASH client were deployed. Each connected link has default 10 Mbps bandwidth allocation. For the implementation of SDN, we used Ryu [24] and OpenFlow v1.3 as southbound APIs.

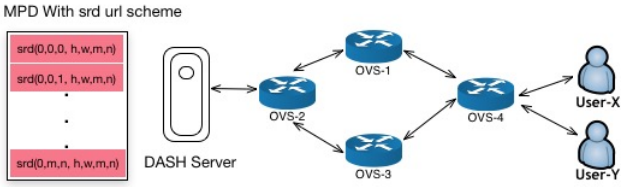


Fig. 7. Mininet SDN topology setup

B. Video Dataset And Media Player

We use MPEG-DASH encoded video datasets to evaluate our models. To test the efficiency of our model, we selected different types of DASH datasets (Table. I) ensuring that each dataset has variations in encoding details.

TABLE I
MPEG-DASH VIDEO DATASET CHARACTERISTICS

Name	Codec	Source Quality	Genre
BigBuck Bunny	H.264/AVC	1080p	Animation

AStream Media player [25] used in [26] is a python based command line tool. It is an emulated video player, which could be used to evaluate the performance of the DASH bitrate adaptation schemes. It can request segments from any multimedia server using MPD file provided to it during the start of the video streaming session. It typical does not provide any GUI for the user to watch the video.

During the video playback, the media player provides logs like Buffer logs and Playback Logs. Buffer logs provide information about Epoch time, Current playback time, current buffer size, current playback state. Playback logs provide information about epoch time, playback time, segment number, segment size, playback bitrate, segment duration and weighted harmonic mean average download bitrate.

C. SDN Traffic Engineering for ROI(s) Traffic Optimization

In this section, we evaluate user's QoE based given ROI bitrates and buffer size. Figure 8 shows that two different ROI regions on a given $2X2$ tile-based video segments. The MPEG-DASH SRD represents the spatial locations, such as tile $\#12$

is represented by SRD syntax value field $\langle 1, 2, 1, 1, 2, 2\rangle$. In our test scenario, ROI-A is covered only by tile #12, while ROI-B is covered by both tile #11 and #12. The number of tiles covering the specific ROI(s) is the number of segments that user's focus at a specific time. At the specific time t, the user will change its ROI, such as from ROI-A to ROI-A' and from ROI-B to ROI-B'. We track the segment bitrate changes for each tile and evaluate how proposed SDN framework can adaptively optimize user's QoE by varying the segments' bitrate based on ROIs' coordination and movement.

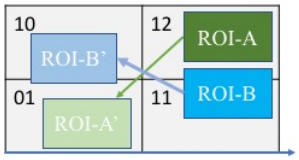


Fig. 8. ROI movement for a $2X2$ tile-based video segments

1) Benchmark with Non-SDN Deployment: To establish the benchmark for comparison purpose, streaming over the traditional network is conducted first. Without any traffic engineering to optimize ROI's bitrate, all the tiles are streaming over one single link. Figure 9(a) and 9(e) depict the bitrate and buffer information for each tile flow. The traffic pattern displays a typical TCP bandwidth allocations among all traffic. The average downloaded bitrate for each tile is $[0.5, 0.4, 0.4, 0.3]$, with an overall average bitrate of 0.4 Mbps. The average buffer size is $[6.7, 6.5, 6.5, 5.7]$, with an overall average buffer of 6.3 units.

2) ROI Bitrate Optimization With SDN Deployed: In this section, we first assume viewer's ROI is on the tile #12, shown in Figure 8. The SDN controller's SRD parser module parses the feedback from the viewer and installs flows on the path [ovs-2, ovs-3, ovs3-4] in Figure 7 for that particular flow. The rest of tiles stayed with the original path [ovs-2, ovs-1, ovs-4]. Figure 9(b) shows that the bitrate rate increases after client's initial setup, which is around ten segments. The average downloaded bitrate for each tile is $[0.4, 0.4, 0.4, 2.0]$, with an overall average bitrate of 0.8 Mbps. The highest bitrate 1.93 is for tile #12, which increases viewer's quality of experience by increasing the bitrate for his/her ROI. Average buffer size in Figure 9(f) is $[6.2, 6.2, 6.2, 6.7]$, with an overall average buffer of 6.3 units. Even though the average buffer is approximately equal to the previous test case, the higher buffer size for ROI tile #12 is increased by 1 buffer unit.

Then we assume viewer's ROI is covered by tile #12 and #11.In this case, ROI and Non-ROI tiles are split equally by two paths. Figure 9(c) shows the average bitrate for ROI tiles' average bitrate $[0.7, 0.9]$, with an overall average bitrate of 0.8 Mbps. The Average buffer size in Figure 9(g) is $[7.7, 7.7]$, with an overall average buffer of 7.7 units. In both metrics, they are better than cases with Non-SDN deployed.

3) ROI Switchover with SDN Deployed: In this test case, we adaptively change viewer's ROI from ROI-A to another ROI-A' shown in Figure 8. The new ROI is still covered by one tile. At the beginning of playback, segments are split by two paths to increase initial ROI's (#12) bitrate. We assume that ROI switches to #01 right after finish streaming the 15-th segment. Figure 9(d) shows the average downloaded bitrate for each tile as $[1.4, 0.4, 0.4, 0.9]$, with an overall average bitrate of 0.8 Mbps. The highest bitrate 1.41 Mbps is for new ROI with tile #01. Previous ROI tile #12 has average bitrate 0.9 Mbps. Average buffer size in Figure 9(h) is $[5.9, 6.2, 6.1, 6.1]$, with an overall average buffer of 6.1 units. Slightly low buffer size compared with one ROI optimization test case is negligible.

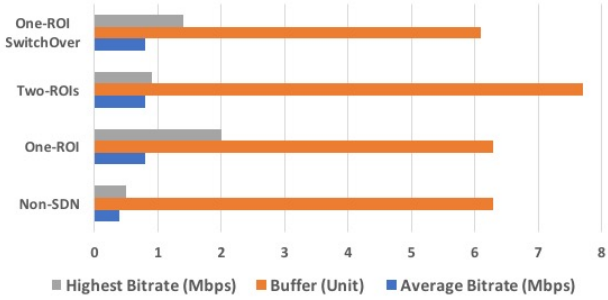


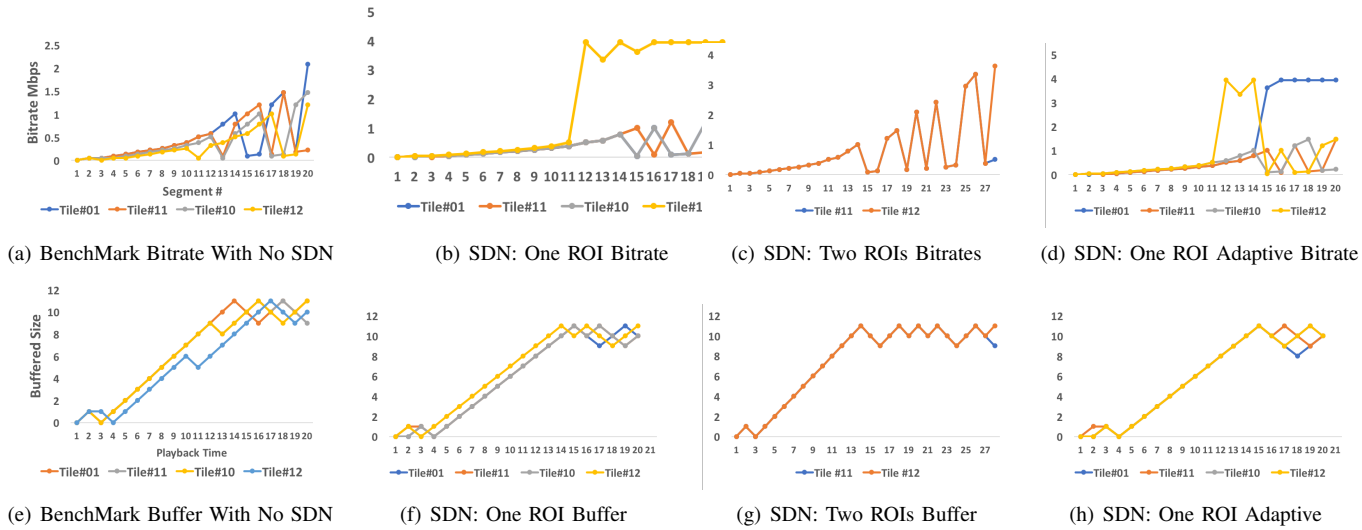
Fig. 10. QoE Improvement Comparison

In summary, from Figure 10, with the SDN-assisted framework, user's QoE can be increased by increasing average ROI's bitrate in both fixed ROI and ROI switchover cases. In our test cases, the increased bitrate is around 100% and buffer size 6%.

VI. Conclusion and Future Work

In this work, we first summarize the difference in immersive content streaming between traditional- and SDN-based network and introduce an SDN-based framework to assist tile-based immersive content streaming. Our goal was to develop an SDN-assisted framework to improve user QoEs for streaming new 3D immersive media such as Virtual Reality and 360-degree video using DASH without changing the underlying design of DASH client itself. We proposed a general application interface regarding a traffic flow alteration mechanism. By optimizing bitrate for viewer's ROI, our proposed framework reduces the overall transferred data size and increase bitrates for ROI to improve the overall viewing experience. The presented framework outperformed compared to the traditional end-client adaptation model. Our model is evaluated and tested on a number of ROI schemes. In all tests, we observed a significant ROI bitrate and buffer increase which signifies improvement of QoE of video streaming.

As future work, we plan to integrate tile-based streaming related issues such as a synchronization problem for different tiles. Because each tile is essentially a separate TCP connection, each tile might have different playback timeline due to package delay in a congested network. We are also planning to include video traffic classification using data signature based dynamic machine learning models and explore additional traffic engineering (TE) methods as well as larger platforms



(a) BenchMark Bitrate With No SDN (b) SDN: One ROI Bitrate (c) SDN: Two ROIs Bitrates (d) SDN: One ROI Adaptive Bitrate

(e) BenchMark Buffer With No SDN (f) SDN: One ROI Buffer (g) SDN: Two ROIs Buffer (h) SDN: One ROI Adaptive

Fig. 9. Bitrate and Buffered Size for Both Non-SDN and SDN deployment

for our approach. New transport protocols such as WebSocket can also be investigated for multiple ROIs switchover at the same time in a single duplex network flow for better network efficiency.

REFERENCES

[1] T. Stockhammer, "Dynamic adaptive streaming over http–: standards and design principles," in *Proc. of 2nd ACM conference on Multimedia systems*, 2011.

[2] S. Liu, X. Xu, S. Lei, and K. Jou, "Overview of hevc extensions on screen content coding," *APSIPA Transactions on Signal and Information Processing*, vol. 4, p. e10, 2015.

[3] C. Concolato, J. Le Feuvre, F. Denoual, F. Maze, N. Ouedraogo, and J. Taquet, "Adaptive streaming of hevc tiled videos using mpeg-dash," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[4] M. Hosseini and V. Swaminathan, "Adaptive 360 vr video streaming based on mpeg-dash srd," in *Multimedia (ISM), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 407–408.

[5] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*. ACM, 2016, pp. 1–6.

[6] A. TaghaviNasrabadi, A. Mahzari, J. D. Beshay, and R. Prakash, "Adaptive 360-degree video streaming using layered video coding," in *Virtual Reality (VR), 2017 IEEE*. IEEE, 2017, pp. 347–348.

[7] S. Y. Lim, J. M. Seok, J. Seo, and T. G. Kim, "Tiled panoramic video transmission system based on mpeg-dash," in *Information and Communication Technology Convergence (ICTC), 2015 International Conference on*. IEEE, 2015, pp. 719–721.

[8] J. Le Feuvre and C. Concolato, "Tiled-based adaptive streaming using mpeg-dash," in *Proceedings of the 7th International Conference on Multimedia Systems*. ACM, 2016, p. 41.

[9] M. Hosseini, "View-aware tile-based adaptations in 360 virtual reality video streaming," in *Virtual Reality (VR), 2017 IEEE*. IEEE, 2017, pp. 423–424.

[10] O. A. Niamut, E. Thomas, L. D'Acunto, C. Concolato, F. Denoual, and S. Y. Lim, "Mpeg dash srd: spatial relationship description," in *Proceedings of the 7th International Conference on Multimedia Systems*. ACM, 2016, p. 5.

[11] T. Stockhammer, P. Fröjdh, I. Sodagar, and S. Rhyu, "Information technologympeg systems technologiespart 6: Dynamic adaptive streaming over http (dash)," *ISO/IEC, MPEG Draft International Standard*, 2011.

[12] T. El-Ganainy and M. Hefeeda, "Streaming virtual reality content," *arXiv preprint arXiv:1612.08350*, 2016.

[13] G. Cheung, Z. Liu, Z. Ma, and J. Z. Tan, "Multi-stream switching for interactive virtual reality video streaming," *arXiv preprint arXiv:1703.09090*, 2017.

[14] A. Mavlankar, P. Agrawal, D. Pang, S. Halawa, N.-M. Cheung, and B. Girod, "An interactive region-of-interest video streaming system for online lecture viewing," in *Packet Video Workshop (PV), 2010 18th International*. IEEE, 2010, pp. 64–71.

[15] N. Q. M. Khiem, G. Ravindra, and W. T. Ooi, "Adaptive encoding of zoomable video streams based on user access pattern," *Signal Processing: Image Communication*, vol. 27, no. 4, pp. 360–377, 2012.

[16] M. Prins, O. Niamut, R. van Brandenburg, J.-F. Macq, P. Rondao Alface, and N. Verzijp, "A hybrid architecture for delivery of panoramic video," in *Proceedings of the 11th european conference on Interactive TV and video*. ACM, 2013, pp. 99–106.

[17] H. Wang, V.-T. Nguyen, W. T. Ooi, and M. C. Chan, "Mixing tile resolutions in tiled video: A perceptual quality assessment," in *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop*. ACM, 2014, p. 25.

[18] W. Min, H. Hannu, J. Pettersson, and Y. Timner, "Optimization of fairness for http adaptive streaming with network assistance in lte mobile systems," in *Vehicular Technology Conference (VTC Fall), 2014 IEEE 80th*. IEEE, 2014, pp. 1–5.

[19] J. J. Quinlan, A. H. Zahran, and C. J. Sreenan, "Datasets for avc (h. 264) and hevc (h. 265) evaluation of dynamic adaptive streaming over http (dash)," in *Proceedings of the 7th International Conference on Multimedia Systems*. ACM, 2016, p. 51.

[20] C. Mueller, "Mpeg-dash in a nutshell." [Online]. Available: https://bitmovin.com/dynamic-adaptive-streaming-http-mpeg-dash/

[21] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[22] "Mininet." [Online]. Available: http://mininet.org/

[23] O. VSWITCH, "Open vswitch," 2013.

[24] S. Ryu, "Framework," 2013.

[25] AStream. [Online]. Available: https://github.com/pari685/AStream

[26] P. Juluri, V. Tamarapalli, and D. Medhi, "SARA: Segment aware rate adaptation algorithm for dynamic adaptive streaming over http," in *2015 IEEE International Conference on Communication Workshop (ICCW)*.