공학석사 학위논문

RL based ABR Streaming for 360 Degree Video

360도 비디오를 위한 강화학습기반 ABR 스트리밍

2022년 08월

서울대학교 대학원

컴퓨터공학부

전 서 호

RL based ABR Streaming for 360 Degree Video

지도 교수 권 태 경

이 논문을 공학석사 학위논문으로 제출함
2022년 08월

서울대학교 대학원
컴퓨터공학부
전 서 호

전서호의 공학석사 학위논문을 인준함
2022년 08월

위 원 장	문봉기	(인)
부위원장	권태경	(인)
위 원	이영기	(인)

ABSTRACT

RL based ABR Streaming for 360-Degree Video

Seoho Jeon
Department of Computer Science & Engineering
The Graduate School
Seoul National University

As Virtual Reality (VR) becomes a more popular media platform, streaming for VR becomes important. Since the 360-degree video does not display the entire frame, the video frame is divided into small sections, *tiles*, and only the tiles to be displayed on the user screen are transmitted. However, current techniques do not reflex their focusing area, such as far left side or top-right corner. This leads to lower users' Quality of Experience.

In this thesis, we introduce an Object Location-Based Adaptive Bitrate Streaming algorithm (OLB) to redeem Field of View prediction errors and define quality functions for Quality of Experience using a Reinforcement Learning technique. The conducted experiments with network datasets demonstrate that our Object Location-Based Adaptive Bitrate Streaming (ABR) achieves better results over the current state-of-the-art models in 360 video streaming.

Keywords: bitrate adaptation, video streaming, reinforcement learning, quality of the experience, tile-based video streaming

Student Number: 2020-23491

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Virtual Reality (VR) is an artificially created environment where users feel like they are engaging with their surroundings. It is used in various fields such as education, architecture, and medical care, beyond entertainment. As the use of VR rises, streaming 360-degree video for VR became more prevalent.

Unlike a regular video, 360 video contains image frames in all angles and directions, but not all the frames are displayed in the users'screen. Thus, the video frame is divided into smaller sections, *tiles*, and only the tiles to be displayed on the user screen are transmitted and streamed to users[31]. This lowers bandwidth usage and improves the Quality of Experience (QoE).

Selecting tiles for video streaming is up to the users'client. The client needs to predict the users'next viewport and request specific tiles of the video chunk. However, this streaming algorithm faces two challenges. First of all, no algorithms can accurately tell which part of the screen the user will watch next. Secondly, although a user is not equally interested in all display screen parts, the current streaming method does not reflect the user's interest level in each tile.

In this study, we propose a novel object location-based adaptive bitrate streaming algorithm to redeem Field of View (FoV) prediction errors and define novel quality functions for Quality of Experience (QoE). We first analyze video contents. This information is used to learn an Adaptive Bitrate Streaming (ABR) algorithm that captures both the FoV prediction result and content feature.

We organize this paper as follows. In Chapter 2, we briefly review adaptive bitrate streaming, 360-degree video streaming, and human peripheral vision. We formulate a 360-degree adaptive bitrate streaming problem with traditional Quality of Experience, in Chapter 3. Then, our Object Location-Based ABR mode is presented in Chapter 4. Performance of our model is evaluated in Chapter 5, along with details on the experiment process. Finally, in Chapter 6, we summarize our study and limitations of our research.

Chapter 2

Background

Video is a sequence of images, also called *frames*, captured electronically. To download or to receive a video from a server, a client requests a fragment of video,*video chunk* (or *video segment*), from the video server. Once the request is accepted by the server, the video will be transmitted from the server to the client; this transmission of video data is referred to as *video streaming.*

Video streaming is typically done over a computer network, and people generally cannot visualize the active transmission with their eyesight. Although there is no obvious spacial limitation where people can see, video streaming has a limit with amount of data it can transfer in a fixed time frame. This maximum amount of data transmitted over a given time is referred as *bandwidth.*

Unfortunately, the amount of a bandwidth is not static, and it fluctuates during the video data transmission per video chunk. Thus, when the client request a video chunk from the server, it is required to send the server an optimal information transferring speed (*bitrate*), typically measured in kilobytes per second (*kbps*), as well.

In this chapter, we introduce a popular bitrate optimization technique Adaptive Bitrate Streaming (ABR) algorithms in Section 2.1. Then, Section 2.2 presents a feedback equation used in ABR. 360-video streaming technology is described in Section 2.3. Lastly, a person's vision and visual range will be listed in Section 2.4.

2.1 Adaptive Bitrate Streaming

ABR is a technique used to stream various types of multimedia, over the internet with adjusted optimal bitrate to provide for a better video streaming experience. Here are three conditions, required for video to be streaming smoothly without rebuffering in a high quality:

High Bitrate	Higher resolution requires more data in a given time to display, which means it requires higher bitrate.
No Empty Buffer	The buffer is a queue of video data waiting to be displayed. Users typically experience video freeze (also called buffering) when the their buffer is empty. Checking the status of the buffer is important to keep the pace of data transfer.
Quick Responsiveness	This is generic condition which applied to a bandwidth size and a change in user request. If the bandwidth is lowered, bitrate also needs be lowered, too. Also, if user suddenly changed the resolution of video or paused a video, bitrate should be adjusted accordingly.

Description 2.1: Video Streaming Conditions

As seen above in Description 2.1, there are two major items, bitrate and buffer, which can be used to calculate optimal bitrates[18, 2, 14]. Method, using previous and current bitrate, are called *throughput-based* ABR methods. It computes correlation between previous and current throughput for the next bitrate. Whereas a *buffer-based* method assumes that the current buffer's occupancy is related to the video rate and current buffer level to decided the next bitrate for the future video chunk.

However, each method only focuses on one condition, and have their own downside. For instance, throughput-based[15] results rely on previous and current bitrate and does not count the fluctuating network's bandwidth size. Thus, the optimized result might be too big for the changed bandwidth or the result wouldn't be optimal if bandwidth got bigger.

On the other hand, the buffer-based method is heavily dependent on the current status of the buffer. Suppose the current buffer is empty, and video is frozen. Then a buffer-based method calculations may not work or would not be able to compute a highest possible bitrate for the next video chunk[12].

In other word, if the bandwidth is fixed size or the status of client's buffer is always non-zero, throughput and buffer-based ABR find the optimal bitrate and provide sustainable video streaming.

2.2 Quality of Experience

Quality of Experience (QoE) is the overall acceptability of an application or service, as perceived subjectively by the end-user[6]. It is used as a feedback signal in both *throughput* and *buffer* based Adaptive Bitrate Streaming altorithm and contributed to a user's perceived quality of service [25].

Let N be the total number of video chunk. Following is a notation and an equation for a QoE in video streaming :

$$QoE = \sum_{n=1}^{N} q(C_n) - \sum_{n=1}^{N-1} \mid q(C_{n+1}) - q(C_n) \mid - \sum_{n=1}^{N} T_n(1) \tag{2.1}$$

C_n	n^{th} video chunck
T_n	Stalling of C_n
$q(x)$	bitrate of given x

We can see that above equation 2.1 can be broken down into three sub-equations, $\sum_{n=1}^{N} q(C_n)$ for video bitrate, $\sum_{n=1}^{N-1} \mid q(C_{n+1}) - q(C_n) \mid$ represents smoothness, and stalling is denoted as $\sum_{n=1}^{N} T_n(1)$. In this equation, smoothness is the bitrate difference between the past chunk and the current chunk, and stalling is the video download latency.

However, equation 2.1 does not count the bitrate difference among tiles in the same video chunk. 360-degree video streaming [8] uses a modified version of the QoE equation, shown below:

$$\begin{aligned} QoE = \sum_{n=1}^{N} q(C_n) - \sum_{n=1}^{N}\sum_{m=1}^{M} \mid q(C_n) - q(C_{n,m}) \mid *p_{n,m} \\ - \sum_{n=1}^{N-1} \mid q(C_{n+1}) - q(C_n) \mid -\mu \sum_{n=1}^{N} T_n) \end{aligned} \tag{2.2}$$

$$q(C_n) = \frac{\sum_{m=1}^{M} q(C_{n,m}) * p_{n,m}}{\sum_{m=1}^{M} p_{n,m}} \tag{2.3}$$

In this equation 2.2, M is a number of tiles in a video chunk, $C_{n,m}$ represents m^{th} tile in n^{th} Video chunk. Also, $p_{n,m}$ is boolean type viewport factor where it will be 1 if $C_{n,m}$ is in the viewport, otherwise 0.

2.3 360 Video Streaming

360 *Video* is video with image data in all directions; 360 video streaming gives users a more natural and smooth view of the video, especially while in motion.

However, a client does not need the full video frames, since the user would not use all the data frames simultaneously. The video server provides video chunks as *tiles*, a part of 360-degree images and video data partitioned by direction. Thus, the client must know the specific tile of the video chunk where the user will watch next to request server for the data.

This prediction for the next viewport tiles is done by a Field of View (FoV) prediction model. Once the client predicts the next viewpoint tile, it requests the server either the specific tiles with high bitrate and other portions with low bitrate or requests only predicted viewport tiles. To support this system, a video server extracts video into tiles and stores video as tiles. The 360-video streaming system architecture is illustrated in Figure 2.1.

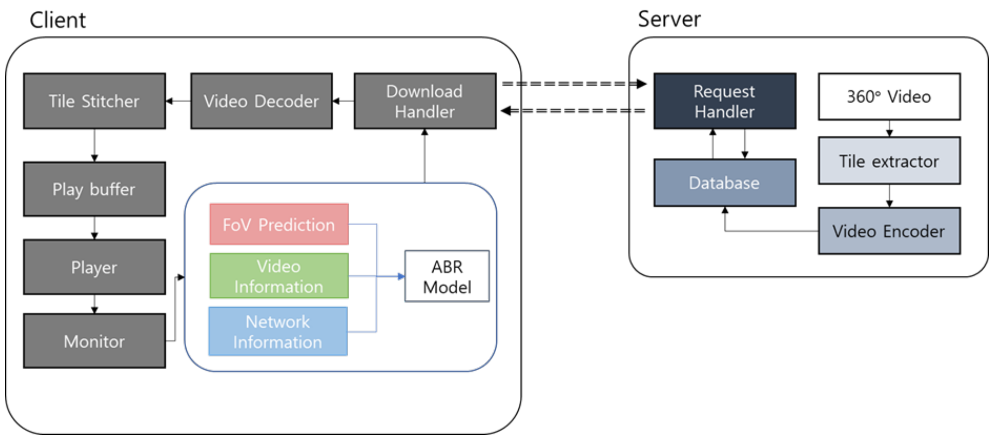


Figure 2.1: Tile-based 360-degree Video Streaming Design

2.3.1 Tile-based Video Streaming

Tile-base streaming is a 360 video streaming technique which sends a partial segment of a video chunk (*tile*), instead of the video chunk with a whole 360-degree view. The methods encode the chunk with K bitrate level, slice the chunk into M tiles, and then, it select tiles to set a bitrate.

Typically, all the selected tills holds same bitrate, like 1 or non-zero value, to differentiate from the rest of non-select tiles which are set to different bitrate, such as 0. Yet, there are algorithms such as ATRIA[22] and 360SRL[8] which calculate and assign different bitrate per tiles.

By splitting the chunk into multiple tiles, a client is allowed to curtail throughput usage. On the other hand, the cost of video transmission is increased, since they need to encode the data to cut into small tiles, and need M tiles to decoded later - once the selected data is transmitted.

Thus, to minimize the cost, some methods merge adjacent tiles with the same bitrate[33] to reduce the number of tiles, or merge entire selected tiles. Then, they predict the viewport with minimum size[35].

In addition, PAAS[31] uses two queues, one for predicted viewport tiles the and other for non-predicted viewport tiles, to optimize the predicted viewport tiles only request. And, as NAS[32], Dasari et al.[5] uses a super-resolution module to improve video quality.

2.3.2 Field of View Prediction

To predict the next user viewport, the client uses past user behaviors or video content features. Depending on the input, the FoV algorithms are classified as follows:

Trajectory-based[3, 7, 13, 21, 11, 35]: Model uses user behaviors to predict the next user viewport with an assumption that a user has one's own behavior pattern. Yet, the user's interest may change over time; the model does not reflect the change promptly.

Content-based[33]: Algorithm predicts the next user viewpoint based on the video content's features. However, it does not ensure personal interest as some features may not relate or reflect users'interest.

Hybrid[4]: The model that use both user behaviors and video content features. This logical background stems from the assumption that if one does not watch a viewport with a behavior pattern, one will watch the video portion that has its own feature.

2.4 Human Peripheral Vision

Carl Gutwin et al.[10] distinguishes 5 levels in the human visual usage: central vision, para-central vision, near peripheral, mid-peripheral, and far peripheral. Each horizontal range is about 2.5°, 4°, 30°, 60°, and 100° to 110° far from the visual axis. Respectively the first three ranges are called the central visual field, and others called peripheral vision [26]. The human can recognize very detailed information in the central vision and color in near peripheral.

Chapter 3

Quality Function for QoE

Due to the limitations of traditional Quality of Experience (QoE), we propose a novel quality function that will be able to consider user interest. Every tile in the port exerts an equal influence on traditional QoE. According to Carl Gutwin et al.[10], the human perceives an object differently depending on where the object is in one's sight. In [26], the visual acuity at the boundary between the central visual field and peripheral vision is only one-sixteenth of foveal value which is the center of the gaze, and visual acuity at out-boundary of para-central vision is one-fourth of foveal value. Thought most effective visual encoding method is position and the second on is color[16], the human cannot distinguish color at peripheral vision.

To demarcate user perceived quality, we define visual importance for each visual range. We set the maximum visual importance to 1 and the minimum to $\frac{1}{5}$. The visual importance of peripheral vision is set to minimum value, for humans cannot recognize the most effective visual encoding method in this range. Then, we map the visual acuity to visual importance using the square root function. Each vision range's visual importance is as follows: visual importance = “central vision”: 1, “para-central vision”: 0.5, “near peripheral” : 0.25, “mid-peripheral”: 0.2, “far peripheral”: 0.2 Using this importance, we design a quality function that factors different influences on QoE depending on the visual importance of each tile. The quality function is as follows:

$$q(C_n) = \frac{\sum_{m=1}^{M} q(C_{n,m}) * (w_1 p_{n,m} + w_2 vp_{n,m})}{\sum_{m=1}^{M} (w_1 p_{n,m} + w_2 vp_{n,m})} \tag{3.1}$$

The notes used to value this are organized in 4.1. Using two different weight parameters, we modify the importance of the viewpoint and the viewport. w_1 $(0, 1]$ is the visual importance of the viewport and w_2 $(0, 1]$ is the visual importance of the viewpoint. High w_1 means standardized tile qualities and high w_2 implies that the human has great interest on one’s viewport.

Notation	Description
N	Total Number of video chuncks ($n \leq N$)
M	Total Number of tiles in a video chunck ($m \leq M$)
$q(x)$	Bitrate of x
C_n	n^{th} video chunck (Current video chunk)
$C_{n,m}$	m^{th} tile in n^{th} video chunk
Tn	Stalling of n^{th} video chunk
$p_{n,m}$	Boolean value representing the precent of (n, m) in ViewPoint. 1 : m^{th} tile in n^{th} video chunk is in the viewpoint 0 : it is NOT in the viewpoint
$vp_{n,m}$	Boolean value representing a viewpoint status. 1 : m^{th} tile in n^{th} video chunk is a viewpoint 0 : is NOT a viewpoint
w_1	weight 1: visual importance of viewport
w_2	weight 2: visual importance of viewpoint

Table 3.1: Variables for the QoE Equation

Chapter 4

Object Location-Based Adaptive Bitrate Streaming

Since Field of View (FoV) prediction models cannot guarantee perfect accuracy, Adaptive Bitrate Streaming (ABR) algorithms need to estimate optimal bitrate with additional data to redeem the FoV prediction errors. We can expect that if one does not watch a viewport with one's behavior pattern, one may watch the video portion that has its feature. So, using video features may help to improve the algorithm. From this assumption, PARIMA[4] reduces the FoV prediction error by using object information in video. We propose an ABR streaming algorithm using image information to make up for the errors.

To capture FoV prediction result and video contents feature, we detect locations of an object in a video frame. We create an object location list $OL = [OL^{(1)},\ OL^{(2)},\ ...,\ OL^{(N)}]$ for a video, where $OL^{(i)}$ is a set of object locations in a i^{th} video chunk and N is the number of chunk in a video. Then, we transform object location ($OL^{(i)}$) to tile location ($m \in M$).

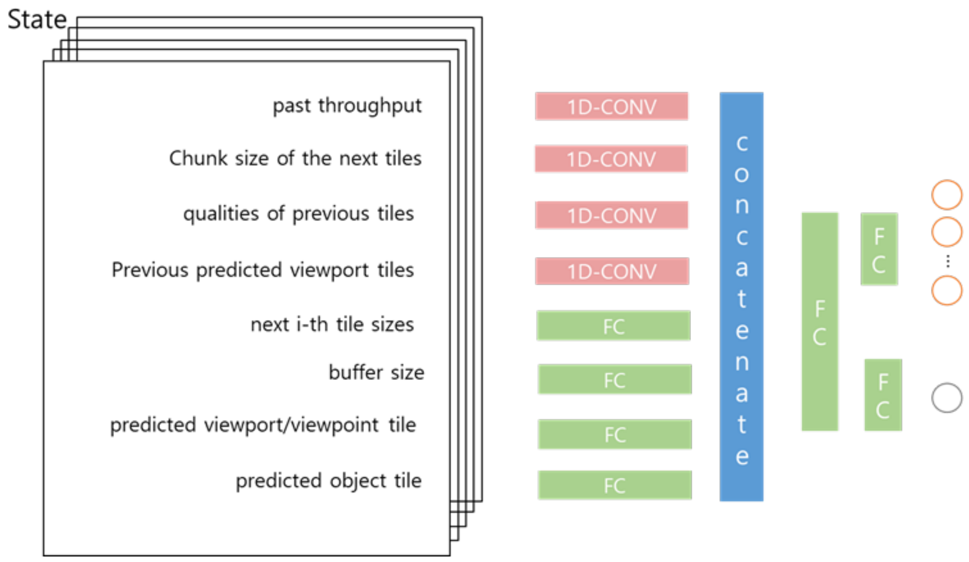


Figure 4.1: Architecture of Object Location-Based Adaptive Bitrate Streaming

We trained a ABR algorithm named Object Location-Based ABR (OLB) with the list OL. We use Advantage Actor-Critic [19] algorithm to train our model. The model maximizes the cumulative reward $R_n = \sum_{k=0}^{\infty} \gamma^k r_{n+k}$ over the training, where γ is a discount factor. The architecture of our model OLB is illustrated in Figure 4.1. and the components of the reinforcement learning algorithm in our model are as below described.

Action

Action is one of the main key components of Reinforcement Learning. It decides what to do next. In ABR streaming, action is mapped with bitrate and search space cost of action is number of possible bitrate K. The action policy aims to choose a bitrate which maximizes QoE. Due to the space cost K required to make a decision, choosing a bitrate for M tiles takes space cost K^M. This space cost is non–polynomial and is currently unable to implement and run in real-time. Therefore, we use one action space model which re-uses the model for all tiles in a video, like ATRIA[22] and 360SRL[8] did.

In step n with state S_n, M states are informed, and the model goes to the next step after finishing all computing for the M states. The action in OLB is defined as $a_{n,m}$ which defines a bitrate for m^{th} tile in n^{th} video chunk.

State

State is information used to decide what happens next. It is a function of history. Even though using the entire client information seems to guarantee high performance, it causes a huge space cost which we cannot build or causes a learning delay that is unable to finish learning in polynomial time. So, in this work, we selected eight inputs for state. The state of OLB is defined as follow:

$$S_n = \{S_{n+1,1},\ S_{n+1,2},\ ...,\ S_{n+1,M}\}$$

$$S_{n+1,m} = \{\overrightarrow{pT_n},\ \overrightarrow{ts_{n+1}},\ \overrightarrow{q_n},\ \overrightarrow{vp_n},\ ts_{n+1,m},\ B,\ vp_{n,m},\ o_{n,m}\}$$

The variables used in the above equations are described in Table 4.1. We use past throughput and buffer size to estimate network state. Past quality is given to minimize the smoothness penalty and past viewpoint is provided to inform past video decisions. Finally, video size, viewport/viewpoint information, and object information are provided to choose which tile to allocate high bitrate.

Notation	Description
$S_{n,m}$	State at step n which choose bitrate for y^{th} tile.
B	Current Buffer Size
$\overrightarrow{pT_n}$	Past Throughputs
$\overrightarrow{q_n}$	[$q(C_{n,1}),\ q(C_{n,2}),\ ...,\ q(C_{n,M})$]
$ts_{n,m}$	Stalling of m^{th} tile in n^{th} video chunk
$\overrightarrow{ts_n}$	[$ts_{n,1},\ ts_{n,2},\ ts_{n,M}$]
$p_{n,m}$	Value which represents the presence of (n, m) in ViewPoint. 0.5: m^{th} tile in n^{th} video chunk is in the viewpoint 0: is NOT in the viewpoint
$vp_{n,m}$	Boolean value which representing a viewpoint status. 1 : m^{th} tile in n^{th} video chunk is a viewpoint $p_{x,y}$: is NOT a viewpoint
$\overrightarrow{vp_n}$	[$vp_{n,1},\ vp_{n,2},\ vp_{n,M}$]
$O_{n.m}$	Boolean value: 1 if m in $OL'^{(n)}$. Otherwise, 0

Table 4.1: Notations for the QoE Equation

Reward

Reward is scalar feedback which signifies how well the agent is doing at step. The agent maximizes the policy's expected cumulative reward $E[\sum_{k=0}^{\infty} \gamma^k r_{n+k}]$. We use the QoE from Equation 2.2 with the quality Equation 3.1 as the reward r which is represented as follows:

$$r_n = q(C_n) - Sm_n - \sum_{m=1}^{M} \mid q(C_n) - q(C_{n,m}) \mid * p_{n,m} - \mu T_n) \tag{4.1}$$

$$Sm_n = \begin{cases} 0, & n < 2 \\ \mid q(C_n) - q(C_{n-1}) \mid, & \text{otherwise} \end{cases} \tag{4.2}$$

For the given Equation 4.1, the first term $q(C_n)$ is the average bitrate of n^{th} video chunk and the second term Sm_n describes smoothness which mirrors the difference between the n^{th} video chunk bitrate and $n-1^{th}$ video chunk bitrate. It is 0 if n is smaller than 2. The next term stands for smoothness which reflects the difference of bitrate among tiles in the n^{th} video chunk. The last term signifies downloading time of n^{th} video chunk.

Chapter 5

Experiments

In this Chapter, we perform experiments on real-world datasets. We compare the performance of the proposed model, Object Location-Based ABR (OLB), with other 360 video Adaptive Bitrate Streaming (ABR) algorithms. Further, we analyze the results on different videos.

5.1 Experimental Setup

We setup a video server simulation environment by benchmarking the environment of Pensieve[17]. We extract a video chunk into 24 tiles with 4 rows and 6 columns as DRL360[34] did and use 6 bitrate levels: 1, 2.5, 5, 8, 16, 35 Mbps. We are encoding video chunk length 1 second, and setting the number of tiles M value as 24 and all the possible bitrate K to be 6.

As we divide frame columns into six parts, one tile covers 60 degrees which covers all of the central visual field. Therefore, set viewpoint weight w_2 to central vision's visual importance 1 and viewport weight w_1 to peripheral vision' s visual importance 0.2. We make FoV prediction dataset using PARIMA[4].

To encode these videos in six difference bitrate level, we encoded videos in *.yuv* format using the ffmpeg[27] command. Then, we extracted these videos to 4x6 tiles using the kvazaar[29] command and encoded them in six bitrate levels using the mp4box[9] command.

5.2 Datasets

In this experiment, we use network trace, video, and viewpoint trajectory dataset to setup simulation environment.

- **Network trace**: we use two public datasets from Ghent[28] and FCC[24]. Ghent[28] is a 4G dataset which recorded throughput in almost every 1000 milliseconds. FCC[24] is a 3G dataset, from December, 2021.

- **Videos and viewpoint trajectories**: We select two videos as a test set from Wu et al[30], a movie clip and a scene from a football match. Each video is encoded with six different bit rates: 1, 2.5, 5, 8, 16, and 35 $Mbps$, then split temporally in one-second chunks. For each video dataset, 48 viewpoint trajectories are attached, where each trajectory is information tracing the viewpoint of a user, see Table 5.1 for more detailed information.

	Movie	Football
Category	Movie	Sport
Length	294s	165s
Chunk Size	1s	1s
Frame Per Second	30	25
Viewpoint Trajectory#	48	48

Table 5.1: The Video Dataset Information

5.3 Baselines

We compare our OLB model with the two 360 video ABR streaming algorithms as below. For fair comparison, we train these models with the proposed QoE for this test.

- **DRL360**[34]: The reinforcement learning based hybrid ABR algorithm which predicts next throughput and uses this result as an input state element instead of past bandwidth.
- **ATRIA**[22]: The reinforcement learning based hybrid ABR algorithm that chooses bitrate for each tile. This algorithm chooses one bitrate for a tile at once.

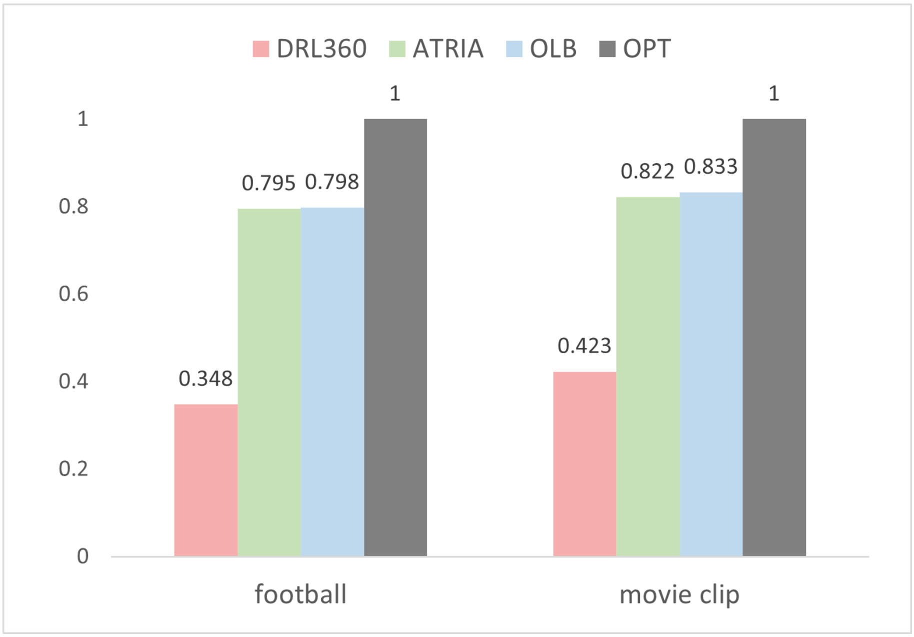
Model	DRL 360	ATRIA	OLB (ours)
Chunk ID	○	-	-
Time Stamp	○	-	-
Buffer occupancy	○	○	○
Size of the tiles of the next chunk	○	○	○
Size of the tile to determine the bitrate	-	-	○
Predicted viewpoint of the next chunk	○	○	○
Past throughput	-	○	○
Estimated bandwidth	○	-	-
Past bitrate	-	○	○
Past download time	-	○	-
Number of Segments remaining	-	○	-
Decided tile qualities	-	○	-
Previous viewport tile	-	-	○
Object location	-	-	○

Table 5.2: Evaluation Models' State Input

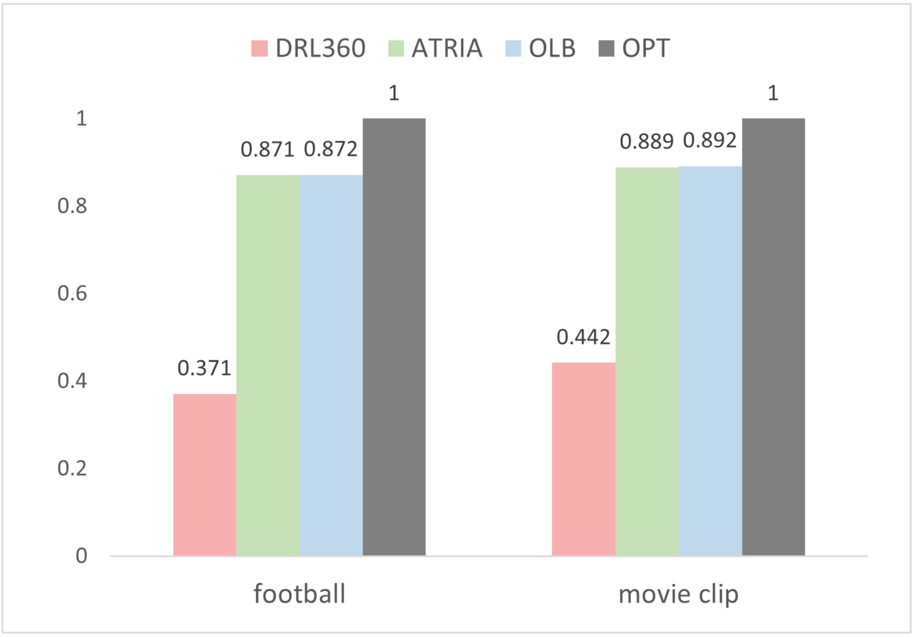
5.4 Performance Evaluations

Finally, we calculated the average QoE of our Object Location-Based ABR (OLB) Algorithm, along with the baseline DRL360 and ATRIA methods. As a reference, we let an optimal QoE, which represent maximum QoE each video can achieve, noted as OPT, and set it to be 1. Below, we show the results of our QoE experiment with existing QoE (Equation 4.1) and our modified quality function, from the Equation 3.1 in Chapter 3.

First of all, we measured the QoE using the existing quality function (Equation 2.3). As shown in the Figure 5.1, there are not much of performance differences between the OLB and ATRIA algorithms with both FCC and Ghent network traces. Nevertheless, the result of DRL360 method is nearly 50% of ATRIA and OLB algorithms.



(a) Average QoE with FCC[24] network trace



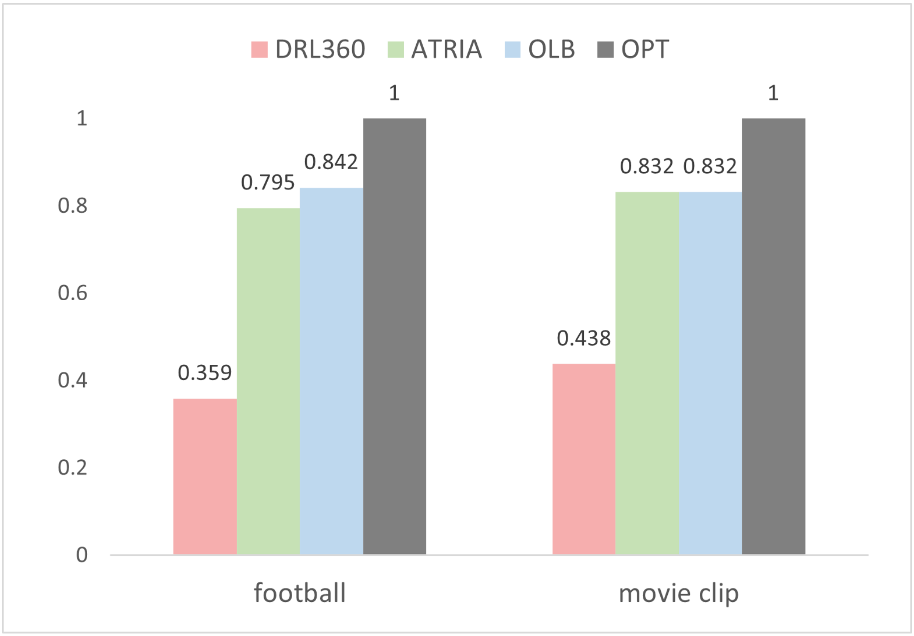
(b) Average QoE with Ghent[28] network trace

Figure 5.1: Average QoE with existing quality function

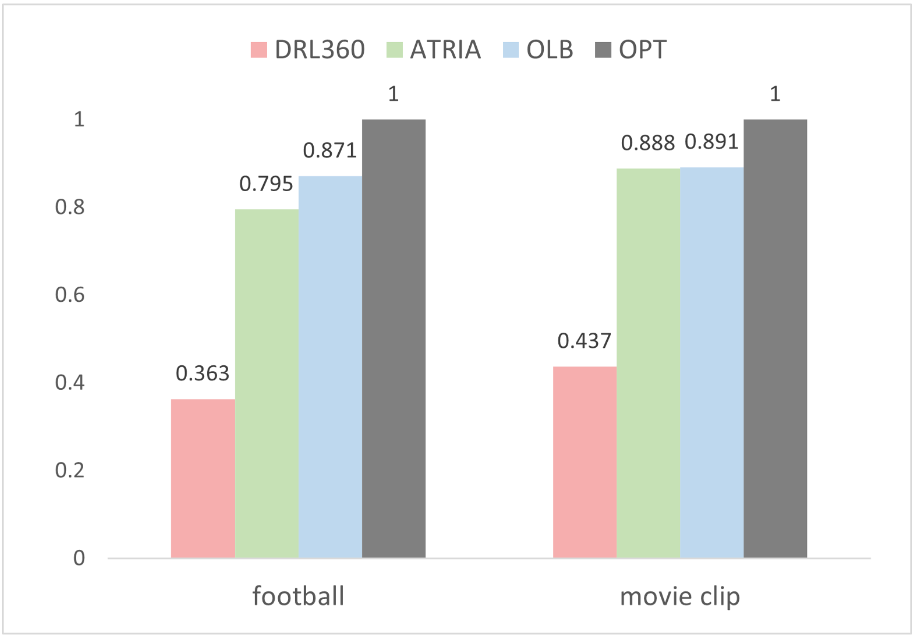
Now, Equation 3.1 accommodates weights of viewport and viewpoint in the QoE calculation. Shwon in Figure 5.2, QoE with OLB and ATRIA algorithms result very close performance, for the movie clip.

But, OLB outperforms ATRIA by 4.7% with FCC and 7.6% with Ghent, using weighted-quality equation, for the football dataset. This is because FoV prediction of football video does not have false positive errors, while movie clip has about $0 - 8\%$ false positive errors.

Overall, DRL360 performs $39.4 - 50.8\%$ lower than OLB in every experiments, regardless of the different video datasets, network traces, and the QoE equation. Although ATRIA and OLB show very similar outcomes when we use the existing quality equation, OLB shows slightly better result with our weighted-quality equation, for the football dataset.



(a) Average QoE with FCC[24] network trace



(b) Average QoE with Ghent[28] network trace

Figure 5.2: Average QoE with new quality function

Chapter 6

Conclusion

In this research, we proposed Object Location-Based ABR (OLB) 360 video streaming algorithm and a revised quality function for 360 video streaming Quality of Experience (QoE). The proposed model is able to capture the video content feature using object location in video and makes up for field of view. In order to optimize user perceived video quality, we analyze the human visual field. Based on this analysis, we design the visual importance factor for each visual field and design a quality function that absotively reflects the quality perceived by the user using this importance factor. The results show the proposed model outperforms other the baseline methods in 360-video streaming task.

Glossary

Abbreviation

ABR	Adaptive Bitrate Streaming
FoV	Field of View
OLB	Object Location-Based ABR
RL	Reinforcement Learning
QoE	Quality of Experience
VR	Virtual Reality

Notation

γ	Discount Factor
B	Current Buffer Size
K	All the Possible Bitrates
K^M	Space Cost for bitrate M
M	Number of Tile per Video Chunk ($m \leq M$)
N	Number of Video Chunks ($n \leq N$)
w_1	Weight for the View-port
w_2	Weight for the ViewPoint

Functions

C_n	n^{th} Video Chunk
$C_{n,m}$	m^{th} Tile in n^{th} Video Chunk
$o_{n,m}$	Value to Indicate If m^{th} Tile of n^{th} Video Chunk is in the Object Location $OL^{(n)}$. Value is set to be 1 for Yes, and 0 for No.
$OL^{(k)}$	Object Location of k^{th} Frame
$p_{n,m}$	Value to Indicate If m^{th} Tile is in the View-port. Value is set to be either 1 or 0.5 for Yes, and 0 for No.
$q(n)$	Bit-rate of n^{th} Video Chunk
$\overrightarrow{pT_n}$	Past Throughput
$\overrightarrow{q_n}$	List of the n^{th} Bit-rates. $[\ q(C_{n,1}),\ q(C_{n,2}),\ ...,\ q(C_{n,M})\ ]$
r_n	An Average Bitrate of n^{th} Video Chunk
R_n	A Cumulative Reward of n^{th} Video Chunk
$S_{n,m}$	State of Step n which chooses bitrate for m^{th} Tile
T_n	Stalling of n^{th} Video Chunk
$\overrightarrow{ts_n}$	$[\ ts_{n,1},\ ts_{n,2},\ ts_{n,M}\ ]$
$ts_{n,m}$	Stalling of m^{th} Tile in n^{th} Video Chunk
$vp_{n,m}$	Value to Indicate If m^{th} Tile is in the View-Point. Value is set to be 1 for Yes, and either 0 or $p_{n,m}$ for No.
$\overrightarrow{vp_n}$	$[\ vp_{n,1},\ vp_{n,2},\ vp_{n,M}\ ]$

Others

$kbps$	Kilobits per Second
$Mbps$	Megabits per second

Bibliography

[1] Barman, N. and Martini, M. G. [2019], 'Qoe modeling for http adaptive video streaming–a survey and open challenges', *IEEE Access* **7**, 30831–30859.
URL: https://doi.org/10.1109/ACCESS.2019.2901778

[2] Bentaleb, A., Taani, B., Begen, A. C., Timmerer, C. and Zimmermann, R. [2019], 'A survey on bitrate adaptation schemes for streaming media over http', *IEEE Communications Surveys Tutorials* **21**(1), 562–585.
URL: https://doi.org/10.1109/COMST.2018.2862938

[3] Chen, X., Tan, T. and Cao, G. [2021], Popularity-aware 360-degree video streaming, IEEE INFOCOM 2021 - IEEE Conference on Computer Communications, pp. 1–10.
URL: https://doi.org/10.1109/INFOCOM42981.2021.9488856

[4] Chopra, L., Chakraborty, S., Mondal, A. and Chakraborty, S. [2021], Parima: Viewport adaptive 360-degree video streaming, *in* 'Proceedings of the Web Conference 2021', pp. 2379–2391.
URL: https://dl.acm.org/doi/10.1145/3442381.3450070

[5] Dasari, M., Bhattacharya, A., Vargas, S., Sahu, P., Balasubramanian, A. and Das, S. R. [2020], Streaming 360-degree videos using super-resolution, *in* 'IEEE INFOCOM 2020 - IEEE Conference on Computer Communications', p. 1977–1986.
URL: https://doi.org/10.1109/INFOCOM41043.2020.9155477

[6] *Definition of Quality of Experience (QoE)* [2007].
URL: https://www.itu.int/md/T05-FG.IPTV-IL-0050/en

[7] Fan, C.-L., Lee, J., Lo, W.-C., Huang, C.-Y., Chen, K.-T. and Hsu, C.-H. [2017], Fixation prediction for 360 video streaming in head-mounted virtual reality, *in* 'Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video', pp. 67–72.
URL: https://doi.org/10.1145/3083165.3083180

[8] Fu, J., Chen, X., Zhang, Z., Wu, S. and Chen, Z. [2019], 360srl: A sequential reinforcement learning approach for abr tile-based 360 video streaming, *in* '2019 IEEE International Conference on Multimedia and Expo (ICME)', pp. 290–295.
URL: https://doi.org/10.1109/ICME.2019.00058

[9] GPA [2020], *MP4Box-1.0.1*.
URL: https://github.com/gpac/gpac/wiki/MP4Box

[10] Gutwin, C., Cockburn, A. and Coveney, A. [2017], Peripheral popout: The influence of visual angle and stimulus intensity on popout effects, *in* 'Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems', p. 208–219.
URL: https://doi.org/10.1145/3025453.3025984

[11] Gŭl, S., Bosse, S., Podborski, D., Schierl, T. and Hellge, C. [2020], *Kalman Filter-based Head Motion Prediction for Cloud-based Mixed Reality*, pp. 3632–3641.

URL: https://doi.org/10.1145/3394171.3413699

[12] Huang, T.-Y., Johari, R., McKeown, N., Trunnell, M. and Watson, M. [2014], 'A buffer-based approach to rate adaptation: Evidence from a large video streaming service', **44**(4), 290–295.

URL: https://doi.org/10.1145/2619239.2626296

[13] Jeong, J.-B., Lee, S., Ryu, I.-W., Le, T. T. and Ryu, E.-S. [2020], Towards viewport-dependent 6dof 360 video tiled streaming for virtual reality systems, *in* 'Proceedings of the 28th ACM International Conference on Multimedia', pp. 3687–3695.

URL: https://dl.acm.org/doi/10.1145/3394171.3413712

[14] Kua, J., Armitage, G. and Branch, P. [2017], 'A survey of rate adaptation techniques for dynamic adaptive streaming over http', *IEEE Communications Surveys Tutorials* **19**(3), 1842–1866.

URL: https://doi.org/10.1109/COMST.2017.2685630

[15] Liu, C., Bouazizi, I. and Gabbouj, M. [2011], Rate adaptation for adaptive http streaming, *in* 'Proceedings of the Second Annual ACM Conference on Multimedia Systems', p. 169–174.

URL: https://doi.org/10.1145/1943552.1943575

[16] MacKenzie, I. S. [2013], *Human-Computer Interaction: An Empirical Research Perspective*, Vol. 48, Morgan Kaufmann Publishers Inc.

URL: https://doi.org/10.5555/2501707

[17] Mao, H., Netravali, R. and Alizadeh, M. [2017], Neural adaptive video streaming with pensieve, *in* 'Proceedings of the Conference of the ACM Special Interest Group on Data Communication', p. 197–210.
URL: https://doi.org/10.1145/3098822.3098843

[18] Meng, Z., Chen, J., Guo, Y., Sun, C., Hu, H. and Xu, M. [2019], Pitree: Practical implementation of abr algorithms using decision trees, *in* 'Proceedings of the 27th ACM International Conference on Multimedia', p. 2431–2439.
URL: https://doi.org/10.1145/3343031.3350866

[19] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Harley, T., Lillicrap, T. P., Silver, D. and Kavukcuoglu, K. [2016], Asynchronous methods for deep reinforcement learning, *in* 'Proceedings of the 33rd International Conference on International Conference on Machine Learning', Vol. 48, p. 1928–1937.
URL: https://doi.org/10.5555/3045390.3045594

[20] Mok, R. K., Chan, E. W., Luo, X. and Chang, R. K. [2011], Inferring the qoe of http video streaming from user-viewing activities, *in* 'Proceedings of the First ACM SIGCOMM Workshop on Measurements up the Stack', p. 31–36.
URL: https://doi.org/10.1145/2018602.2018611

[21] Pang, H., Zhang, C., Wang, F., Liu, J. and Sun, L. [2019], Towards low latency multi-viewpoint 360° interactive video: A multimodal deep reinforcement learning approach, *in* 'IEEE INFOCOM 2019 - IEEE Con-

ference on Computer Communications', pp. 991–999.

URL: https://doi.org/10.1109/INFOCOM.2019.8737395

[22] Park, S., Hoai, M., Bhattacharya, A. and Das, S. R. [2021], Adaptive streaming of 360-degree videos with reinforcement learning, *in* '2021 IEEE Winter Conference on Applications of Computer Vision (WACV)', pp. 1838–1847.

URL: https://doi.org/10.1109/WACV48630.2021.00188

[23] Qian, F., Han, B., Xiao, Q. and Gopalakrishnan, V. [2018], Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices, *in* 'Proceedings of the 24th Annual International Conference on Mobile Computing and Networking', p. 99–114.

URL: https://doi.org/10.1145/3241539.3241565

[24] *Raw Data Arranged by Year - Fixed* [2021].

URL: https://www.fcc.gov/reports-research/reports/measuring-broadband-america/measuring-fixed-broadband-eleventh-report

[25] Reiter, U., Brunnström, K., de Moor, K., Larabi, M.-C., Pereira, M., Pinheiro, A. M., You, J. and Zgank, A. [2014], 'Factors influencing quality of experience', pp. 55–72.

URL: https://hal.archives-ouvertes.fr/hal-01159290

[26] Strasburger, H., Rentschler, I. and Jüttner, M. [2011], 'Peripheral vision and pattern recognition: A review', *Journal of Vision* **11**, 13–13.

URL: https://doi.org/10.1167/11.5.13

[27] Tomar, S. [2006], 'Converting video formats with ffmpeg'.
URL: https://doi.org/10.5555/1134782.1134792

[28] van der Hooft, J., Petrangeli, S., Wauters, T., Huysegems, R., Alface, P. R., Bostoen, T. and De Turck, F. [2016], 'HTTP/2-Based Adaptive Streaming of HEVC Video Over 4G/LTE Networks', *IEEE Communications Letters* **20**(11), 2177–2180.
URL: https://doi.org/10.1109/LCOMM.2016.2601087

[29] Viitanen, M., Koivula, A., Lemmetti, A., Ylă-Outinen, A., Vanne, J. and Hămălăinen, T. D. [2016], *Kvazaar: Open-Source HEVC/H.265 Encoder*.
URL: https://doi.org/10.1145/2964284.2973796

[30] Wu, C., Tan, Z., Wang, Z. and Yang, S. [2017], A dataset for exploring user behaviors in vr spherical video streaming, *in* 'Proceedings of the 8th ACM on Multimedia Systems Conference', pp. 193–198.
URL: https://dl.acm.org/doi/10.1145/3083187.3083210

[31] Wu, C., Wang, Z. and Sun, L. [2021], Paas: A preference-aware deep reinforcement learning approach for 360° video streaming, *in* 'Proceedings of the 31st ACM Workshop on Network and Operating Systems Support for Digital Audio and Video', p. 34–41.
URL: https://doi.org/10.1145/3458306.3460995

[32] Yeo, H., Jung, Y., Kim, J., Shin, J. and Han, D. [2018], Neural adaptive content-aware internet video delivery, *in* 'Proceedings of the 13th USENIX Conference on Operating Systems Design and Implementation', p. 645–661.
URL: https://doi.org/10.1109/INFOCOM41043.2020.9155477

[33] Yu Guan, Chengyuan Zheng, X. Z. Z. G. and Jiang, J. [2019], Pano: optimizing 360° video streaming with a better understanding of quality perception, *in* 'Proceedings of the ACM Special Interest Group on Data Communication', pp. 394–407.
URL: https://doi.org/10.1109/INFOCOM.2018.8486282

[34] Zhang, Y., Zhao, P., Bian, K., Liu, Y., Song, L. and Li, X. [2019], Drl360: 360-degree video streaming with deep reinforcement learning, *in* 'IEEE INFOCOM 2019 - IEEE Conference on Computer Communications', pp. 1252–1260.
URL: https://doi.org/10.1109/INFOCOM.2019.8737361

[35] Zhou, C., Xiao, M. and Liu, Y. [2018], Clustile: Toward minimizing bandwidth in 360-degree video streaming, *in* 'IEEE INFOCOM 2018 - IEEE Conference on Computer Communications', pp. 962–970.
URL: https://doi.org/10.1109/INFOCOM.2018.8486282

초록

360 도 비디오를 위한 강화학습 기반 적응 비트레이트 스트리밍 기법

Virtual Reality (VR) 은 엔터테인먼트를 넘어 교육과 건축, 의료 등 다양한 분야에 사용되고 있다. VR이 보편화 됨에 따라, VR을 위한 360 도 비디오 스트리밍의 중요성이 높아졌다. 360 도 동영상은 동영상의 전체 프레임을 표시하지 않기 때문에 동영상 프레임을 타일로 분할하여 사용자 화면에 표시할 타일만 전송한다. 어떤 타일을 높은 bitrate 으로 전송할지 결정하기 위해 client 나 server 에서 사용자의 다음 시선을 예측한다. 이러한 예측은 때때로 틀리지만 기존 비디오 스트리밍 방법은 예측에만의존하여 의사 결정을 내리고, 사용자가 화면의 어느 부분에 관심을 가지고 있는지 반영하지 않는다. 이는 사용자 QoE를 떨어트린다.

이 연구에서는 사용자 화면 예측 오류를 보완하기 위해 새로운 객체 위치 기반 Adaptive Bitrate Streaming (ABR) 알고리즘인 Object Location-Based ABR (OLB) 를 제안하고 사용자의 체감 퀄리티를 높이는 새로운 평가함수를 정의한다. 실제 네트워크 데이터를 가지고 실험한 결과 360 도 비디오 스트리밍에서 제시한 모델의 성능이 다른 비교 모델들보다 높음을 확인했다.

주요어: 적응형 비트레이트, 비디오 스트리밍, 강화학습, quality of experience, 타일 기반 비디오 스트리밍

학번: 2020-23491