

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO
PORTO



Bandwidth Prediction for Adaptive Video Streaming

Gustavo Manuel Esteves Pelayo

Dissertação realizada no âmbito do Mestrado em Engenharia Eletrotécnica e de
Computadores

Orientador: Professora Maria Teresa Andrade

Abstract

Quality of User Experience (QoE) enhancement in the context of multimedia applications, particularly in Video Streaming, has always been a concern within the media industry and research community. QoE encompasses a number of factors namely, at the physiological level, and technological. One aspect that has been addressed is how to provide increased levels of realism and to achieve that, applications are starting to offer multi-perspective visual content or multi-view. However, this imposes an additional burden on the equipment and transmission channels as the amount of information raises greatly. This, along with variations that typically occur in the available capacity of a best-effort network, can have a deep impact on the video streaming experience. To try and alleviate such a burden on the network resources whilst minimizing service deterioration, Adaptive Bit Rate (ABR) algorithms based on bandwidth can be used. However, accurate bandwidth estimation is not an easy task. Among other factors, accuracy and latency are significant challenges to overcome. As widely proven in the literature, factors that contribute the most to degrading the QoE in video streaming services are re-buffering events that freeze the video playout and introduce "jumps" in the normal video temporal sequence. The main goal of an ABR algorithm is to maximise viewer QoE which translates into maximising the average video quality, ensuring playout continuity, and preventing freezing and jumps. To accomplish this it is necessary that the latency with which the ABR algorithm follows the variations in the network bandwidth is minimal. However, frequent changes in quality are also annoying and thus estimations should be as accurate as possible. The target of an efficient ABR algorithm is to comply with network variability whilst minimising the frequency and duration of rebuffering events.

This thesis has proposed to review the current state-of-the-art active bandwidth estimation tools specially suited for a video streaming system based on HTTP Adaptive Streaming (HAS). The goal was to identify adequate solutions and evaluate their performance in the context of such a video streaming system. The work conducted has enabled to understand the main principles and alternatives of ABR algorithms and contemporary challenges they faced. The objectives were successfully met as an ABR algorithm was effectively integrated and tested in a laboratory prototype. The study conducted also concluded that among the most popular bandwidth estimation tools, the Maximum TCP Throughput based ones proved to be the most suitable.

An analysis of the effect of the bitrate adaptations on user's QoE was conducted through a survey. The study concluded that viewers are extremely sensitive to video quality deterioration specially when it includes frame loss and freezing. Viewers also proved to be more sensitive to bitrate reduction when the video content includes scenes with an abundance of fine details. Videos which presented a person speaking on the foreground, specially, produced the most disturbance to users after bitrate reduction.

Agradecimentos

Queria começar por agradecer à professora Maria Teresa Andrade e ao engenheiro Tiago Soares da Costa por toda a disponibilidade e simpatia ao longo do processo desta dissertação.

Uma vez que entregar tese também significa o fim de um longo (talvez demasiado) percurso como estudante, queria aproveitar para agradecer, de uma forma mais genérica, a toda a gente que me acompanhou até aqui:

À minha mãe, ao meu pai e à minha madrinha por terem estado comigo sempre nas alturas mais alegres e me ajudarem a sair das alturas mais difíceis, que acabaram por moldar quem eu sou hoje.

Ao meu irmão Gonçalo, o meu companheiro diário, ao meu irmão Guilherme, o meu terceiro orientador de tese, orientador de curso e da vida no geral, por me terem sempre acompanhado com a paciência que só um irmão consegue.

A todos os meus amigos e colegas: ao Zé Diogo pelo companheirismo ao longo do meu percurso académico (10^o ano até ao último ano da faculdade), ao Ivan, por todas as experiências e aventuras por que passámos, ao Escaleira por todas as conversas, corridas e maluqueiras e a todos os amigos que me acompanharam.

Queria agradecer em especial à minha namorada, a Mariana, porque no final do dia é ela que me atura, e não é uma tarefa fácil.

"Video Killed the Radio Star."
- The Buggles

Contents

List of Figures	10
List of Tables	13
1 Introduction	16
2 Context	18
2.1 Multi-View	18
2.2 SmoothMV	20
2.2.1 Hot&Cold Matrix	20
2.2.2 Server Core	22
2.2.3 Client Core	22
2.2.4 MPEG-DASH and Multi-View	23
3 Literature Review	24
3.1 Media Streaming Protocols	24
3.1.1 Adaptive Streaming	24
3.1.2 MPEG-DASH	26
3.2 Media Presentation Description	27
3.3 Video Codecs	27
3.3.1 Lossy vs Lossless Encoding	29
3.3.2 H.264/AVC vs H265.HECV	29
3.4 Bandwidth Estimation	30
3.5 Defining Bandwidth Estimation	31
3.5.1 Capacity	31
3.5.2 Available Bandwidth	32
3.5.3 Transmission Control Protocol (TCP) Throughput and Bulk Trans- fer Capacity	33
3.6 Active Bandwidth Estimation vs Passive Bandwidth Estimation	34
3.6.1 Active Bandwidth Estimation	34
3.6.2 Passive Bandwidth Estimation	36
3.6.3 Model-based Bandwidth Estimation	37
3.6.4 Final Remark	37
3.7 How Video Stream Quality Impacts Viewer Behavior	37
3.7.1 View-level Stream Quality Metrics	38
3.7.2 Metrics for Viewer Behavior	38
3.7.3 Correlation vs Causation	38
3.7.4 Statistical Power and Statistical Relevance	39
3.7.5 Their Findings	40
3.8 Abrupt Bitrate Reduction vs Smooth Bitrate Reduction	43
3.9 Perception of Quality and Video Content	43
3.10 Related Works	44
3.10.1 Adaptive Bitrate Streaming	44
3.10.2 Client Based Bitrate Adaptation	45
3.11 Server-Based Adaptation	46
3.12 Network-Based Adaptation	47
3.12.1 QDASH - Quality of Experience (QoE)-aware Dynamic Adaptive Streaming over HTTP (DASH)	47

4	Methodology	49
4.1	System Breakdown	49
4.1.1	View Selection Module	49
4.1.2	View Buffering Module	51
4.1.3	Server Core	52
4.2	System Developed for the Bandwidth Estimation	54
4.2.1	SmoothMV Scheme Adaptation	54
4.2.2	Adaptive Bitrate Streaming (Adaptive Bitrate (ABR)) Scheme	54
4.3	Software Development	56
4.3.1	Deploying the Bandwidth Estimation Tools	58
4.4	Bandwidth Estimation Tools	60
4.4.1	Iperf	60
4.4.2	Thrulay	60
4.4.3	Ntttcp	61
4.4.4	Pathrate	61
4.4.5	Pathload - Yaz	61
4.4.6	Initial Gap Increasing (IGI)	61
4.4.7	Packet Transmission Rate (PTR)	62
4.4.8	pathChirp - Assolo	62
4.4.9	Abing	63
4.5	Content Preparation	64
4.5.1	Server Video Preparation	64
4.5.2	Video Content	66
5	Results	68
5.1	Bandwidth Estimation Tool Objective Performance	68
5.1.1	Capacity Estimation Tools	68
5.1.2	Available Bandwidth Estimation Tools	70
5.1.3	TCP Throughput and Bulk Transfer Capacity	73
5.1.4	Tools Performance Classification	79
5.2	User's Quality of Experience	80
5.2.1	Structure of the Survey	80
5.2.2	Video Streaming for different Available Bandwidth	81
5.2.3	Abrupt vs Smooth Quality Reduction	84
5.2.4	Quality Reduction for Different Contents	85
5.2.5	Stutter and Frame Loss caused by Iperf and Ntttcp	87
6	Conclusion	89
6.1	Future Work	89
7	References	91

List of Figures

1	Predictions for Asia Pacific Video Streaming Market for 2020 - 2030. Adapted from [2]	16
2	The Tracking Results from a Multi-View Tracker [9]	19
3	Overall SmoothMV system architecture	20
4	The Hot&Coldmatrix, with its distinct 3 stages	21
5	The Server Core, with its distinct 3 Layers	22
6	Heatmap obtained for the Recurrent Attention Model Dataset [14]	23
7	Broadcasters Answers to the Question "Which video streaming formats are you currently using?" [17]	25
8	. Simple example of dynamic adaptive streaming [18]	26
9	DASH Data Model [19]	27
10	the Discrete Cosine Transformation explores the image's spatial redundancy in order to remove irrelevant data.	29
11	Comparison between CTUs in H264 vs H265. In H.265, the CTU's size is determined by the regional information. Thus, there is much more coding precision in more detailed areas of the image, at the same time discarding data from less relevant regions. [30]	30
12	Pipe model with fluid traffic for 3-hop network [6]	32
13	TCP (and therefore Bulk Transfer Capacity (BTC)) implements five standard congestion control algorithms	33
14	Packet Pair Dispersion as in [36]	35
15	spikes in PDF of packet inter-arrival time indicate a small busy time interval as in [41]	36
16	Minimum number of matched pairs required in a viewer abandonment experiment to detect a given effect size with statistical power at least 80% [46]	40
17	Viewers start to abandon the video if the startup delay exceeds about 2s. Viewers also tend to quit more quickly if the video is longer. [46]	41
18	Viewers start tend to abandon the video much quicker if their internet connection is better. [46]	41
19	Correlation of normalized rebuffer delay with play time, the rebuffer delay was normalized by dividing for the video to rebuffer divided by the total video time. [46]	42
20	The probability of returning within a specified return time is distinctly smaller after a failed visit than after a normal one. [46]	42
21	Individual Mean Opinion Score for the video with max quality is roughly the same than for the gradual quality reduction [46].	43
22	Illustration of quality versus bitrate trade-off as in [54].	44
23	At the same encoding bitrate, varying inter-stream (on the left) and intra-stream (on the right) scene complexity results in varying display qualities, or vice versa as in [54]	44
24	Llama Sample Trace with Base Measurements and Moving Harmonic Mean Plotted [60]	46
25	Basic Server-based Bitrate Adaptation [54]	46
26	Basic Network-based Bitrate Adaptation [54]	47
27	The overall QDASH system's architecture [57]	48
28	Client Side Head Tracking Client	50

29	Log File generated with three instances of logging	50
30	Video Buffering Module Downloading the Initial Segments	51
31	Video Buffering Module Downloading the Segments and sequentially play- ing them	52
32	Server Core Folder Organization	53
33	Overall SmoothMV Scheme	53
34	Adapted SmoothMV Scheme	54
35	Adapted SmoothMV Scheme	55
36	Windows Form designed to show the current Bandwidth Estimation and corresponding Video Quality and Log File generated by the Bandwidth Estimation Layer.	57
37	Updated Head Tracking Graphical User Interface (GUI).	58
38	Adapted SmoothMV Scheme with Virtual Machines	59
39	WSL running Bandwidth Estimation Tool in Ubuntu's Bash shell	59
40	Formula for IGI's Algorithm [68]	62
41	Sequential Server Download	65
42	Sequential Server Download	65
43	Parallel Server Download	66
44	Frame from Sleeping in Space	67
45	Frame from GoPro: Foggy Forest MTB	67
46	Frame from Harry Kane's Penalty	67
47	Quocient between Capacity and Bulk Transfer Capacity in different speed links [38]	69
48	Comparison between all tools without any interference or cross traffic . . .	69
49	Abing Results for Cross-Traffic	71
50	Yaz Results for Cross-Traffic	71
51	IGI Results for Cross-Traffic	71
52	PTR Results for Cross-Traffic	71
53	Comparison between tools response to upload limitation	72
54	Iperf's response to upload limitation	73
55	General Required Bandwidth for Streaming Guidelines [77]	74
56	Iperf Measurements every 5 seconds	74
57	Iperf's response to upload limitation	75
58	Frame Loss resulting from Iperf Estimations	76
59	Ntttcp response to upload limitations	76
60	Frame Loss resulting from Ntttcp Estimations	77
61	Thrulay response to upload limitations	78
62	Frame Loss resulting from Thrulay Estimations	78
63	Video presented to users via Youtube's Link	80
64	Correspondence between Color and Answer for the question "How would you classify the video quality?"	81
65	The average and standard deviation of critical parameters	82
66	Correspondence between Color and Answer for the question "How notice- able were the interruptions in the video?"	83
67	The average and standard deviation of critical parameters	83
68	Correspondence between Color and Answer for the question "How notice- able was the Quality Reduction in the video?"	84
69	The average and standard deviation of critical parameters	84

70	Correspondence between Color and Answer for the question "How would you classify the video quality?" for videos with different content.	85
71	Viewers answers to the Question: "How would you classify the video quality?" for videos with different content.	85
72	Viewers answers to the Question: "How would you classify the video quality?" for Canadian Space Agency's "Sleeping in Space" with highest quality	86
73	Viewers answers to the Question: "How would you classify the video quality?" for videos with different content with lowest quality.	86
74	Viewers answers to the Question: "How would you classify the video quality?" for Canadian Space Agency's "Sleeping in Space" with lowest quality	86
75	Correspondence between Color and Answer for the question "How would you classify the video experience?"	87
76	Viewers answers to the Question: "How would you classify the video experience?" for Canadian Space Agency's "Sleeping in Space" adapted by Ntttcp	87
77	Viewers answers to the Question: "How would you classify the video experience?" for Canadian Space Agency's "Sleeping in Space" adapted by Iperf	88

List of Tables

1	Machines and Network Specifications	56
2	Measurement result of Exp.1	79

Abbreviations and Symbols

ABR Adaptive Bitrate	9
AI Artificial Intelligence	17
ABW Available Bandwidth	32
BW Bandwidth	56
BTC Bulk Transfer Capacity	10
DASH Dynamic Adaptive Streaming over HTTP	8
GUI Graphical User Interface	11
HTML HyperText Markup Language	24
HTTP Hypertext Transfer Protocol	22
IP Internet Protocol	31
OS Operating System	58
QoE Quality of Experience	8
SLoPS Self-Loading Periodic Streams	35
SCTP Stream Control Transmission Protocol	60
TCP Transmission Control Protocol	8
UDP User Datagram Protocol	60
VPS Variable Packet Size	34
VR Virtual Reality	18
WSL Windows Subsystem Linux	59

Chapter 1

1 Introduction

The exponential advances in computing technology, high bandwidth storage devices, and high-speed networks have made On-Demand Video Streaming not only possible but widely available. As a direct consequence, it was observed an explosive growth of the internet and a great demand for multimedia information on the web. Video streaming over the Internet has quickly risen to become the preferred platform for the video visualization experience. The possibility of playing a video segment before the entire video has been transmitted is preferable and fundamentally different from having to wait for a download to complete before playback. The statistics show how streaming has visibly transformed the global media landscape and impacted viewing behavior around the world. According to Statista[1], in 2021, the global number of Video on Demand subscriptions amounted to 1.2 billion, and this number is expected to grow by over 400 million in the following six years.

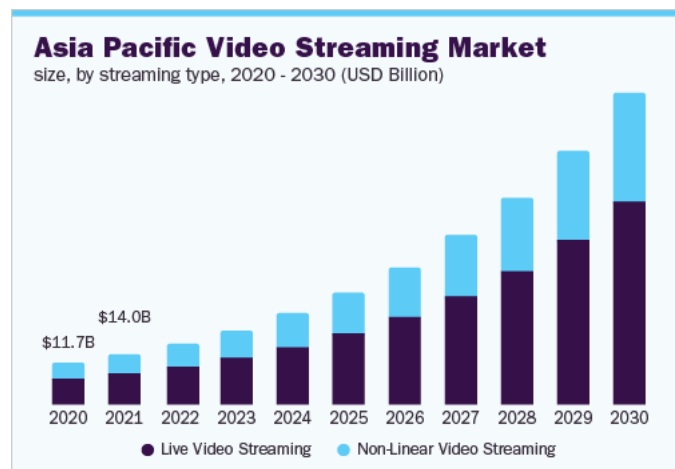


Figure 1: Predictions for Asia Pacific Video Streaming Market for 2020 - 2030. Adapted from [2]

The impact of Video Streaming has brought increased attention from the research community. There's been an increased focus on the enhancement of User Experience. One aspect that has been addressed is how to provide increased levels of realism and to achieve that, a lot of applications are being developed which promise to offer multi-perspective visual content or multi-view. Their advent has provided users the ability to freely navigate within a 3D scene via images captured from multiple cameras. However, there's an inevitable load increase on the equipment and transmission channels as the amount of information increases. To try and alleviate such a burden, multiple strategies are being studied. One area of increasing interest is the analysis of user visual behavior, which has been recently addressed through the use of Artificial Intelligence (AI) techniques. Its relevance is so significant that the market is increasingly looking for datasets to train AI-based algorithms and be able to improve the offer and adapt their products to the user. A visual behavioral dataset has been developed for multi-view content for the training and validation of a neural network. The dataset contains information from head-tracking data and allows the generation of heat maps based on user potential interest in 360° multi-view setting. This thesis's work is focused on the study and development of mechanisms for bandwidth estimation in order to decide the adequate quality/bitrate of the view's perspective corresponding to areas of user interest predicted by the neural network.

This thesis, besides focusing on the SmoothMV multi-view system which is integrated into, also has a more broad analysis on Bandwidth Estimation for On-Demand Video Streaming. The system developed for adapting the quality developed and integrated into SmoothMV only contemplates the streaming of the single view predicted by the neural network. The one view streaming experience, veritably, corresponds to the conventional video streaming scenario. As such, a more general analysis of bandwidth adapted video streaming and the corresponding User's Quality of Experience is presented in this thesis.

A vital indicator of how well a network's QoE is performing is its bandwidth availability. The amount of available bandwidth has a direct impact on how well a multimedia application performs and this is particularly relevant in the case of Video Streaming. For applications with a large data load, managing real-time available bandwidth can have a drastic positive impact on the application performance, as well as the interactive performance, which are particularly sensitive to the expected network variations and problems [3].

The three most commonly categorized approaches for bandwidth estimate are: active or probe-based techniques, passive or sensing-based methods, and analytical or model-based methods. Active probing, in which a few test packets are sent through the link and utilized to determine the network state, is the simplest and most efficient technique for determining the bandwidth that is readily available. As such, the work here presented will only be focused on active probing techniques.

Some previous works have focused on the evaluation of active bandwidth estimations tools [4][5][6]. This particular work aims to expand the information present in the literature, as well as extend the existing knowledge for the multi-view context, with special emphasis on the SmoothMV system.

In this study – the performance of the bandwidth estimation tools are examined in three categories: accuracy, robustness (stability, response time, response to interference) and applicability to the scenario at hand.

Chapter 2

2 Context

2.1 Multi-View

The video service has effectively changed our daily lives dramatically. As a direct consequence of the great technical advances in various areas, different video services have been offered a growing amount of flexibility. The digitization of the television is an example of this. Digital television allow the for the transmission of audio and video signal free of interference and more importantly allows for a much greater spectrum efficiency than the traditional Analog TV. The digital television is a crucial component of the continuing digital revolution that is bringing about the information society since it can smoothly integrate with other communication systems, computer networks, and digital media, enabling datacasting and multimedia interactive services [7].

For most of the video streaming applications, the conventional single -view is the most viable option. However, there's a some limitations if the content provider wants to deliver a more immersive experience. The most obvious one is that it only provides one view direction for an event at any time instance while users may want to watch the scene from different perspectives. Other limitation is the fact that the viewer takes a completely passive role. For example, in a fasted-paced sports event, the audience or even the coach/instructor often want to watch the video from comprehensive views, which may enhance their experience or enable them to form an more accurate assessment of the game [8].

Multi-view presents significant potential to offer interactive and innovative experiences to users. However, some aspects of the multi-view technology, have yet to be properly optimized.

Traditionally, in a multi-view environment, in contrast with 360° and Virtual Reality (VR) applications, there is a limited number of cameras which are sent to client and the missing views are produced on locally. Given the complexity of this view synthesis apparatus, significant research effort has gone into optimizing the trade-off between bandwidth gains and computing resources, with the goal of achieving smooth navigation and viewing quality. Nonetheless, the optimization of navigation interactivity, or how the user informs the system of their choice of new points of view, is still an area that is mostly

unexplored. A novel solution has been proposed that aims to dive into the relatively uncharted field of user's behavioural analysis in multi-view, the SmoothMV system [10].

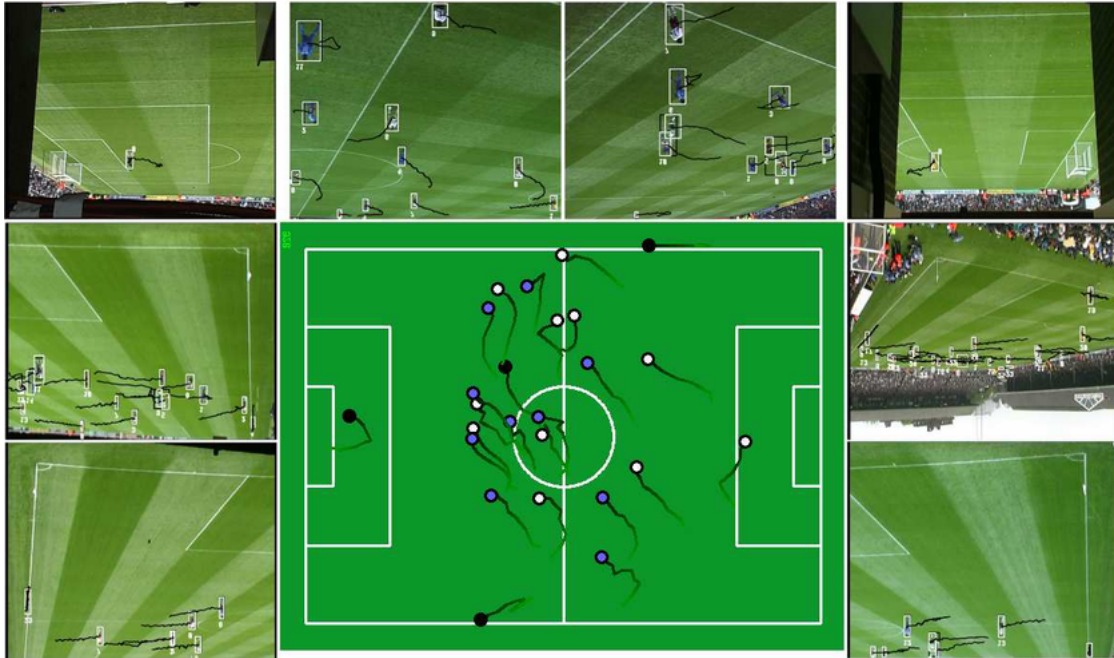


Figure 2: The Tracking Results from a Multi-View Tracker [9]

2.2 SmoothMV

Multiple system have been developed in the past which allowed to navigate multi-view environments. Most of these were based on sub-optimal navigation mediums such as keyboards, pointing devices or joysticks. All These solutions suffer from the same underlying problem, they do not replicate how users interact in real-life environments, leading to a lesser degree of realism. SmoothMV removes the need for the explicit intervention of the user or dedicated equipment. The system aims to improve the User's Quality of Experience and deliver an immersive, real-life experience by providing smooth transitions between views. SmoothMV represents a multi-view system which relies on a non-intrusive head tracking approach to enhance navigation and the viewer's QoE.

The functional architecture of the SmoothMV system can be broken down into the client-server model represented on figure 7.

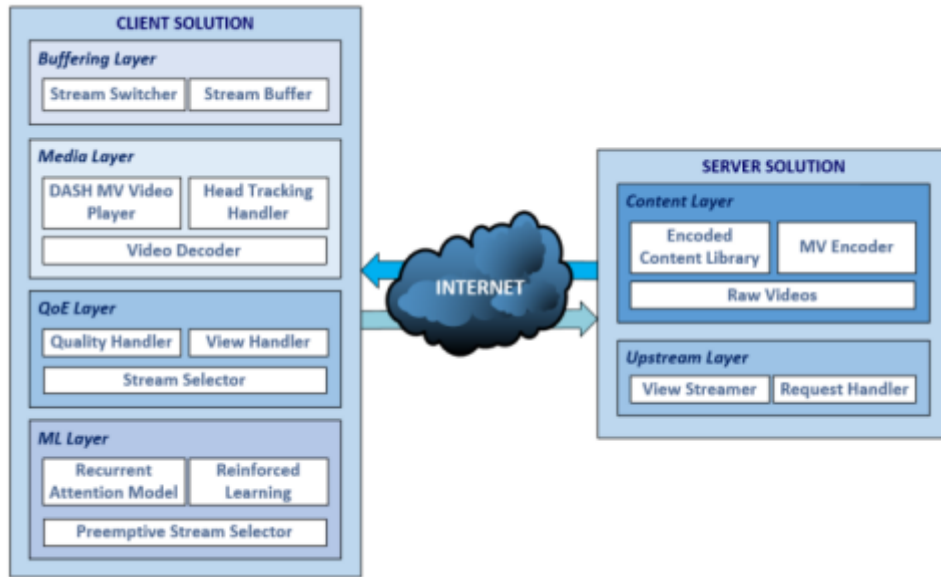


Figure 3: Overall SmoothMV system architecture

The system relies on the on MPEG-DASH norm for the streaming. The server Side is simply responsible for storing and encoding the video. The Client side deals with the procedures required for seamless tracking and view switching in addition to the playback of multi-view content. The tracking is made on available tracking technologies (Intel RealSense [11]), coupled with some custom-designed components.

2.2.1 Hot&Cold Matrix

The SmoothMV's underling central operation is based on the novel concept, the Hot&Cold matrix. The Hot&Cold matrix was conceived as a mechanism to assist the system in identifying the multi-view perspectives to be requested to the server and presented/buffered in

the client. It maps the user viewing behaviour and establishes a relation with the available scene perspectives. The matrix is divided into 3 main regions of different area proportions: Central (11.1%) Intermediate (33.3%) and Peripheral (55.6%) (represented in Figure 4 in different colours).

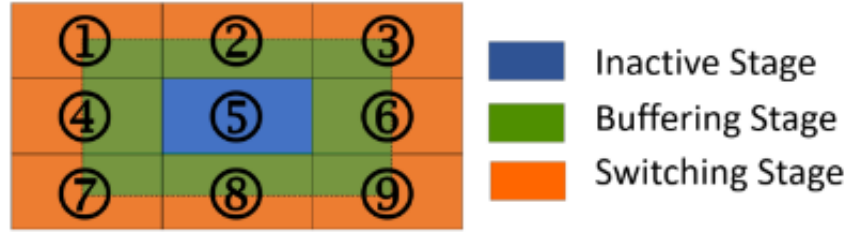


Figure 4: The Hot&Coldmatrix, with its distinct 3 stages

The matrix was constructed based on experiments conducted with 10 users to register their viewing behaviour when watching multi-view videos. Based on the collected tracking data, the attention maps were estimated, leading to the definition of the 2D matrix. Based on the User's head-tracking data, the system maps the focus of the user's attention for the current view into the Hot&Cold matrix. The Intel RealSense framework computes the yaw, pitch and roll head pose data from Euler angles into 2D coordinate space using x, y and z axes values. Euler angles allow the description of the orientation of a rigid body with respect to a fixed coordinate system [4]. By using rotation matrices B, C and D as an example, a general rotation A can be written as:

$$A = BCD \quad (1)$$

Due to the existence of multiple conventions for Euler angles, the x - convention was used for the SmoothMV scenario. With this convention, Euler angles (ϕ, θ, ψ) can be defined through the following sequence of rotations:

- 1st rotation is obtained through angle θ on z-axis, resorting to the D matrix;
- 2nd rotation is conducted by angle θ $[0, \pi]$, on x' (former x-axis), using the C matrix;
- 3rd rotation is achieved by angle ψ on former z-axis), by means of the B matrix.

Using this sequence of rotation, along with the associated x, y and axis, it is possible to accurately pinpoint the head position and orientation within a tridimensional space and translate into the part of the screen the which the user is glancing.

The matrix is divided into 9 sections which correspond to 9 different perspectives of the multi-view scene. The image only needs to be shifted when the user is in the intermediate zone. When it reaches this point the system starts buffering. Once the view switching phase is over, it returns to the inactive stage, since the user starts facing the central part of the screen when the view is switched. The operation of the Hot&Cold matrix ensures

the minimization of the latency as the expected next perspective is already being buffered by the system.

2.2.2 Server Core

The server core focuses on encoding/post-processing tasks, real-time handling of user requests and selection of appropriate versions of available content. It is composed by three functional layers, the Media Handling Layer (Content Layer), the Request Handling layer and the View Streaming Layer.

The Media Handling Layer is responsible for encoding and storing the multi-view videos for multiple qualities. This is done using HEVC and for preparing the content in conformance to MPEG-DASH using the FFMPEG [12] and GPAC [13] software packages. Unfortunately, DASH does not support natively the identification of the different perspectives of multi-view content. Consequently, in order to establish a relationship between viewing angles and cells of the Hot&Cold matrix, a new descriptor file, named View Model, was introduced.

Hypertext Transfer Protocol (HTTP) requests are managed by the Request Handling layer, which also determines which appropriate encoded section needs to be retrieved.

The View Streaming layer uploads the desired segments to the Client using an Apache HTTP server.

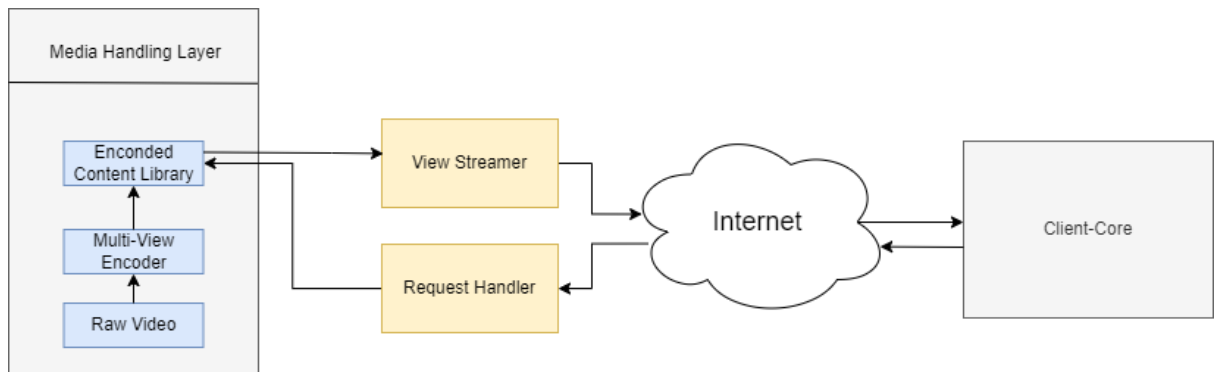


Figure 5: The Server Core, with its distinct 3 Layers

2.2.3 Client Core

The Client Core performs end-user tasks to ensure smooth content playback, notably head tracking, view identification and switching, and QoE assessment.

Firstly, the client core pre-emptively downloads an N number of segments to an initial buffer before starting to playback the downloaded content. After the initial buffer finishes playing the N segments, a M number of segments are continuously downloaded to a player buffer. The M segments are download from a single view if in the user's gaze is in the Inactive stage, or, alternatively, from additional views if in the Buffering stage, using Representations with Lower Quality. Two separate queues are maintained, corresponding to buffered and playback views.

If the user, subsequently, changes to the Switching Stage during a period t , a view switching request is issued. Following $t+1$ segments will be switched between playback-/buffer queues, respectively.

Pre-emptively buffering the probably next view reduces the switching latency and offering different qualities queues reduces the load on the network.

The Client core is split into four layers: Buffering, Media, QoE and ML.

The buffering layer is responsible for downloading and managing the next segments. The QoE layer, takes care of the decision process, concerning the analysis of viewing conditions and perspectives to be buffered/presented, working closely with the buffering layer.

The Media layer decodes and playbacks the video segment into the GPAC MP4Client's DASH Multi-view player and incorporates the Head Tracking Handler module.

The machine learning layer provides an alternative process to determine the most likely view to be required. However the neural network integrated into this layer requires a dataset to train and validate it. The dataset has been, in the meantime, developed[14], nevertheless by the time of the development of this thesis, the Recurrent Attention Model has not yet been completely integrated in the SmoothMV.



Figure 6: Heatmap obtained for the Recurrent Attention Model Dataset [14]

2.2.4 MPEG-DASH and Multi-View

The work presented in SmoothMV utilizes MPEG-DASH norm, as such, it is suitable for any video player. The major setback with utilizing the norm is that it does not support multiple views. Its Spatial Representation Description feature targets specifically the streaming of UHD video and does not provide the means to identify and correlate multiples views. SmoothMV addresses this issue by integrating a novel view management system, allowing dynamic stream selection on-the fly whilst maintaining compatibility with the MPEG-DASH standard. One of the ultimate goals of the SmoothMV's research is to propose an alteration to the DASH standard in order to include multi-view.

Chapter 3

3 Literature Review

3.1 Media Streaming Protocols

Streaming can be defined as the continuous transfer of audio and video data from a server to a client. In video streams, information is transmitted over the internet in a compressed form and is instantly viewed by the viewer. A constant stream of data is used to send the data, which is played as it comes in. The user requires a player, a specialized application that decompresses and delivers audio and video data to speakers and the display.

Since the advent of streaming, several different protocols have been proposed, with varying purposes and efficiencies. The most recognizable participants of these include Adobe's Real-Time Messaging Protocol (RTMP), Apple's HTTP Live Streaming (HLS) protocol, and MPEG-DASH.

The Real-Time Messaging Protocol (RTMP), was the de facto standard for sending video over the internet in its early stages. RTMP is a TCP-based protocol designed to provide persistent, low-latency connections, which in turn facilitates fluid streaming. The protocol started out as the background system behind live and on-demand streaming with Adobe Flash Player. The Flash plugin powered 98% of internet browsers[16] in its prime and RTMP was used ubiquitously.

RTMP, much like most norms, allows streaming by breaking data into small packets often referred to as chunks. RTMP also supports combining multiple streams into a single connection. The Protocol is used to stream multimedia data between Flash Media Server and Flash Player. However, since Adobe announced the end of Adobe Flash, its use has been decreasing drastically. HyperText Markup Language (HTML)-based technologies have been since steadily taking the main bulk of the streaming market [17]. This new category of streaming protocols utilize HTTP based web servers, typically stored on Content Delivery Networks (CDNs), to transmit the video chunks at an adaptive rate.

3.1.1 Adaptive Streaming

The most popular HTTP based protocols are Apple's HLS and MPEG's DASH. HLS is first of the two and is largely the most dominant. The Moving Pictures Expert Group (MPEG) developed the DASH, as an open-source standard alternative to Apple's HTTP Live Streaming (HLS) protocol. Of the two adaptive streaming protocols MPEG-DASH has the best chance of becoming the unifying standard. Both DASH and HLS follow the same paradigm. The material to transmitted is divided into time chunks that are provided

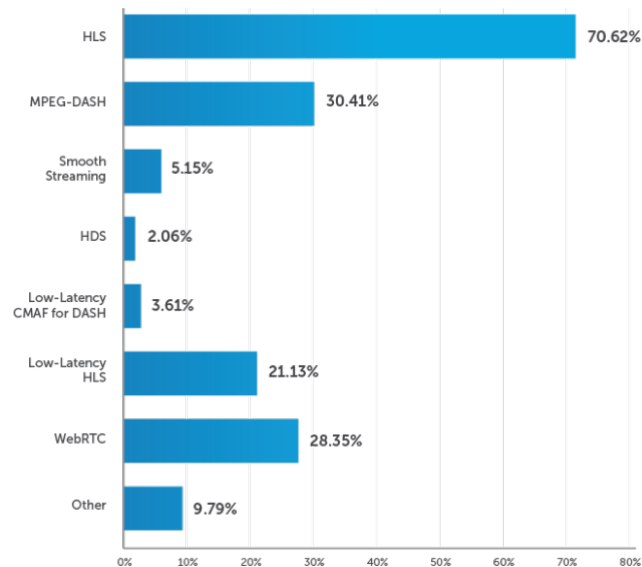


Figure 7: Broadcasters Answers to the Question "Which video streaming formats are you currently using?" [17]

at the user's request and typically last between 2 and 10 seconds. These requests are sent in a way that ensures real-time audiovisual content reproduction while also taking into account network restrictions.

Adaptive bitrate streaming adjusts video quality based on network conditions to improve video streaming over HTTP networks. This process makes playback as smooth as possible for viewers regardless of their device, location, or Internet speed. In the case of MPEG-DASH, the video source is encoded at different alternative bitrates to fit a variety of network conditions. Then, based on factors like bandwidth and device type, the video player selects the highest-quality file that the device can play with the smallest amount of buffering possible. This allows playback to be as smooth as possible for end-users, regardless of the device or Internet speed.

The process is usually started by a device requesting segments of the video bitstream at the highest available quality. It may happen that after streaming the first segments of video and audio and monitoring the effective network bandwidth, the device realizes that the actual available bandwidth is lower than what the highest quality demands. As such, it is necessary to switch the bitstream to lower quality to maintain continuous playback. The device continues playing the content at these rates until the network bandwidth increases and then it switches the video up to a higher quality.

The choice for DASH between the DASH and HLS for SmoothMV and, consequently, the worked developed in this thesis, can be justified by the fact that MPEG-DASH specification is the most recent one and yet with more penetration and support. The standard also is completely open-source and cross-platform which help validating the data from the system across multiple machines.

3.1.2 MPEG-DASH

DASH, Dynamic Adaptive Streaming over HTTP, is an HTTP-based streaming method in which the video data is encoded in different chunks of smaller sizes and those chunks are encoded at different quality levels (bitrates). This allows the video to be streamed at different quality levels and it allows the quality to be switched in the middle of the playback.

The main steps of the MPEG-DASH process can be simplified as:

- **Encoding and Segmentation on the video** - The server decomposes the video into different smaller parts of a few seconds of length and encodes these chunks into a standard encoding format so that they can be interpreted by different devices. The server keeps an index list of the contents for the video segments;
- **Delivery** - When the video playback is started by the user, the encoded video chunks are sent via the internet to the client's device.
- **Decoding and Playback** - As the data is streamed to the user, the data is decoded and played back. The quality of the video is adapted to the bandwidth available at each moment.

The multimedia content is first and foremost captured and stored on an HTTP server and is delivered using HTTP. The content exists on the server in two parts: Media Presentation Description (MPD), and Segments, which contain the actual video in form of chunks.

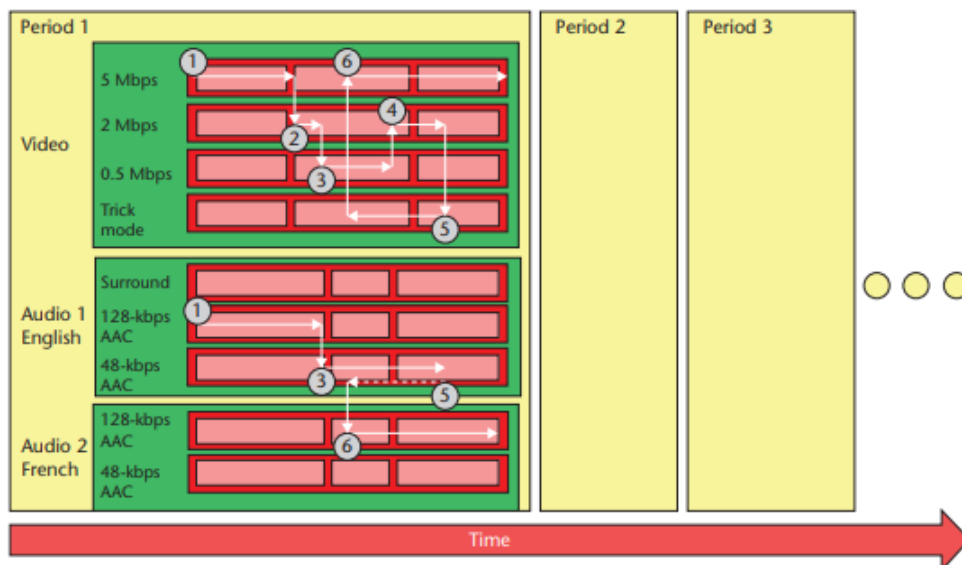


Figure 8: . Simple example of dynamic adaptive streaming [18]

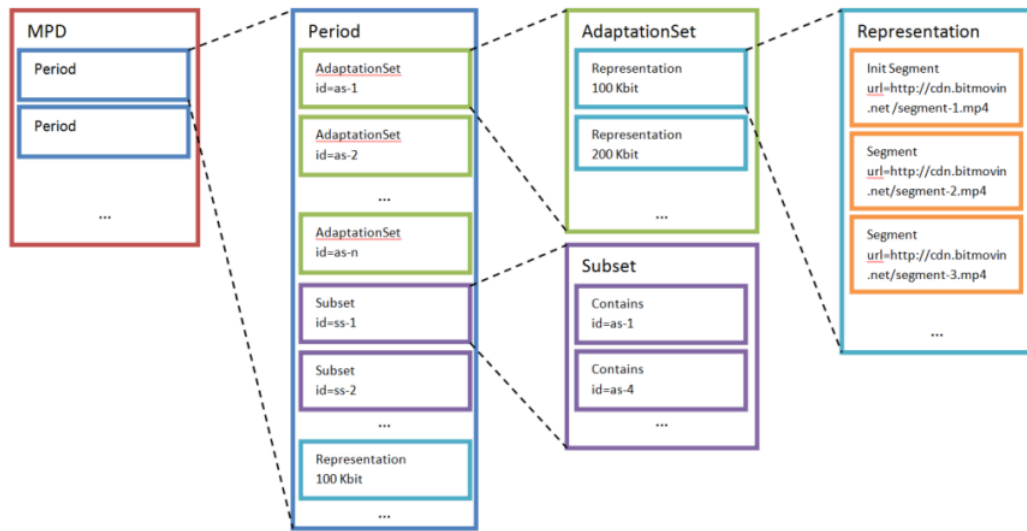


Figure 9: DASH Data Model [19]

3.2 Media Presentation Description

Through the MPD, the DASH client learns about the program timing, media-content availability, media types, resolutions, minimum and maximum bandwidths, the various encoded alternatives of multimedia components, accessibility features and required digital rights management, media-component locations on the network, and other content characteristics [18]. The DASH client then selects the appropriate encoded alternative and starts streaming the content by fetching the segments using HTTP GET requests.

As it was previously mentioned, in the case of the SmoothMV system, the base of the MPD file has been extended to include information about the views, more specifically the location of the views in relation to the central view. Thus, once the information on the focus of attention has been predicted by the neural network it is possible to consult the MPD to know which URL should be used (the URL that corresponds to the view that was predicted). Several URLs are presented for each view for different qualities/compression levels.

One important aspect to consider is that In DASH systems, in order to have smooth playback, video segments are buffered before being displayed. However video buffer mechanism becomes more complicated for multi-view interactive video streaming. Long delay for switching the view is generally not acceptable during the streaming process and the user must be able to freely switch the view. Thus, it is necessary for the viewpoint to be ready when the user so needs it and that it is where the SmoothMV system and the neural network output play a defining role in User’s experience.

3.3 Video Codecs

Similar to other multimedia data types like picture and video, video also needs effective channels for collecting, storage, and delivery. Video appears to be a particularly challenging format to encode. The challenge of data compression is made more difficult by video since it concurrently provides several pictures and audio files. In addition, the medium often has strict requirements for low latency and high bandwidth.

The video encoding process can be defined as the task of transforming a RAW video file into a digital compressed file. Before being encoded, the video is stored as a simple sequence of images. The process ends with a fluid video, much smaller in size than the initial format. As noted by the Wowza official website: "In order to compress the raw video into a more manageable size, encoders use video and audio codecs, which apply algorithms to shrink the bulky video for delivery. To put it more simply: encoding describes the process of compression, whereas codecs describe the means for doing so." [20]

Codecs are made of two essential components: an encoder to compress the files, and a decoder to decompress them. There are codecs for data, audio, image and video. In the case of video, several codecs have been developed that aim to respond to the demands imposed by the medium's particularities, namely H.264/AVC [22], H.265/HEVC (High Efficiency Video Coding)[23], AV1[24], and more recently proposed, H.266/VVC[25]. When working on a system which involves encoding video, one comes inevitably confronted with the question of which of these to utilize.

3.3.1 Lossy vs Lossless Encoding

Much of the visual and audio information presented in video can be effectively removed without altering the user perception. When compressing data, sometimes it is preferable to lose some of the original information to achieve a greater compression.

Lossy compression is a technique in which data is purposely discarded during the compression. Redundant and unnecessary information can be removed without affecting the perceptual original content after being decompressed. Many techniques have been developed to achieve these effects, which explore temporal, visual and spectral redundancy.

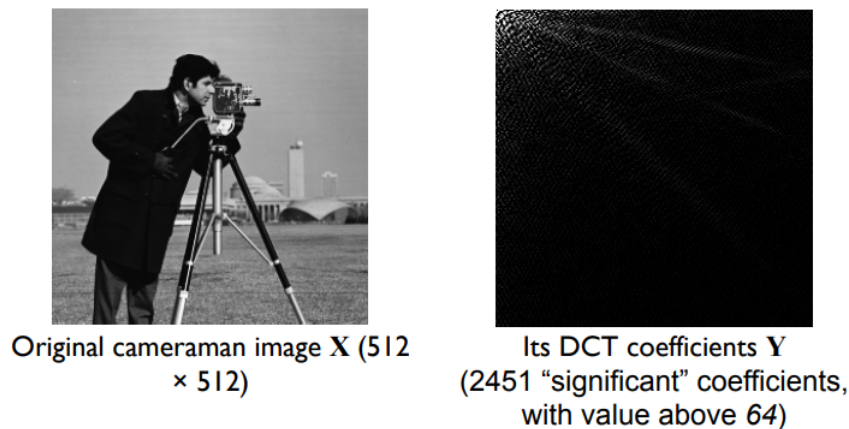


Figure 10: the Discrete Cosine Transformation explores the image's spatial redundancy in order to remove irrelevant data.

When it is preferable to not lose any type of information, it is called lossless encoding. Compressing a group of images into a .zip file without compromising the data integrity can be considered lossless encoding.

In the context of video encoding, the lossy option is mostly used due to the size of the raw video data and the unsuitability of it. Video Codecs deploy lossy techniques to reduce the size of the video files while at the same time maintaining the most important aspects of the video for visual perception.

3.3.2 H.264/AVC vs H265.HECV

HEVC presents itself as evolution of the AVC codec and offers a great deal of improvements relating to the previous iteration. Nevertheless, H.264 continues to be widely used in most streaming services [28], mostly due to the codec's speed, balance of video quality/size and widespread hardware and software support.

H.265 offers much greater compression efficiency and allows the support of 8k video. It also generates files with much smaller sizes than AVC, thus decreasing the bandwidth necessary for streaming. This makes it an ideal codec for high-resolution streaming. [20]

What accounts mostly for the difference in the video compression, is how each of the standards processes frames. H.265 processes information in what are known as "Coding Tree Units" ,CTUs, in contrast to H.264 macroblocks. Information can be compressed more effectively with CTUs since they can process up to 64x64 blocks in comparison to the 4x4 to 16x16 block sizes that macroblocks can handle. [29]

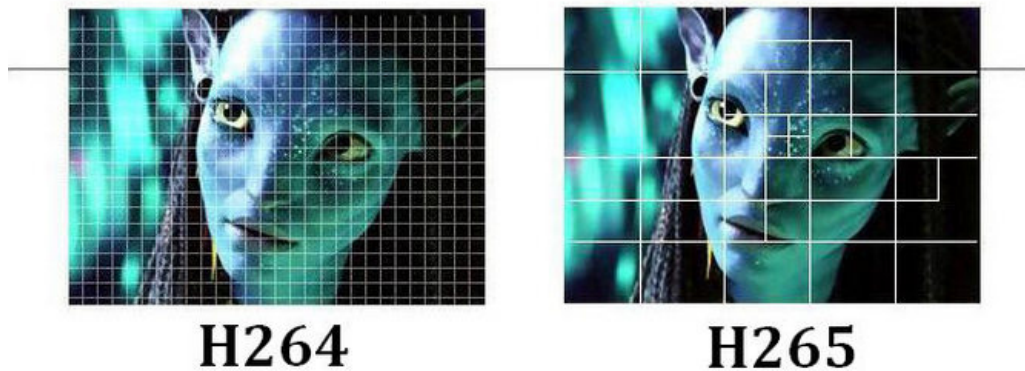


Figure 11: Comparison between CTUs in H264 vs H265. In H.265, the CTU's size is determined by the regional information. Thus, there is much more coding precision in more detailed areas of the image, at the same time discarding data from less relevant regions. [30]

The MPEG group developed the **MCV!** (MCV!) [26] specification, presenting itself as an extension to H.264. Similar multi-view adaptation have also been proposed for H.265 [27]. Several studies have been made in order to evaluate the performance of the H265 for multiview streaming[31][32][33], proving its validity and efficiency. All the videos which were used in this work were encoded using the HEVC codec.

3.4 Bandwidth Estimation

One of the most import metrics for evaluating the performance of a network link is the available bandwidth. In this context, the link can be as a physical or logical connection between two entities at various layers of the protocol stack.

With the increased need for application level traffic optimization, the bandwidth estimation has become very valuable. This is specially true for DASH based streaming applications. To make DASH adaptable to network conditions, video encoding information is stored in the Media Presentation Description (MPD), from which a client can select a suitable quality of video segment for downloading with the current network state. Most adaptive streaming platforms have algorithms implemented for adapting the bitrate through analysis of the bandwidth. These are called Adaptive Bandwidth Streaming Algorithms.

ABR algorithms calculate the available bandwidth through different techniques, some approximate the bandwidth by measuring the packets throughput while other work by measuring bandwidth at the network level. In the following section some of the most popular techniques for bandwidth estimation will be presented as well as some definition related to bandwidth.

Estimation of the available bandwidth is a very challenging task. This is can be attributes to end to-end delay system variances and the dynamics of the Internet. Various techniques have been developed in the past in order to to accurately estimate bandwidth with variable levels of success. It is import to note that the success of a tool may not be necessarily determined by its ability to accurately estimate the bandwidth. Different techniques have different end goals regarding the estimation, some aim for stability, other aim for precision, etc. The very definition on bandwidth can vary among the different tools.

3.5 Defining Bandwidth Estimation

Currently there a lot of different definitions of bandwidth estimation present in the literature. Some concepts are used interchangeably to express the measured bandwidth, such as pre-hop end-to-end capacity, available bandwidth, throughput and bulk transfer. Additionally, there is also some confusion on what the technique applied in measuring. In [34] the author claims that under the self-loading periodic streams methodology, the available capacity is measure, on the other hand, in [35], the author defines that the self-loading periodic streams evaluates the available bandwidth. In an attempt to to make sense of the different concepts, in the following section it is presented a definition of each of what the student interprets as each these metric.

3.5.1 Capacity

Physical Capacity, is the theoretical maximum amount of data that the link can support. Hence, it allows to distinction between links at data link layer and at the Internet Protocol (IP) layer. At theInternet Protocol (IP)layer a hop delivers a lower rate than its nominal transmission rate due to the overhead of layer 2 encapsulation and frames. Supposing the nominal capacity of a segment is C_{L2} , the time to transmit a packet of size L_{L3} in theIPlayer with a layer 2 overhead of H_{L2} is:

$$\Delta_{L3} = \frac{E_{L3} + H_{L2}}{C_{L2}} \quad (2)$$

So the capacity of that segment at theIPlayer is:

$$C_{L3} = \frac{L_{L3}}{\Delta_{L3}} = \frac{L_{L3}}{\frac{E_{L3} + H_{L2}}{C_{L2}}} = C_{L2} \frac{1}{1 + \frac{H_{L2}}{L_{L3}}} \quad (3)$$

We can use the definition pointed by the authors in [36]: the capacity of a hop is the maximum possibleIPlayer transfer rate at that hop. From equation (2) the maximum transfer rate at theIPlayer results from MTU (Maximum Transmission Unit)-sized packets. So we define the capacity of a hop as the bit rate, measured at theIPlayer, at which the hop can transfer MTU-sizedIPpackets.

In the network layer. we can define capacity as: *the capacity of an end-to-end path is the maximum IP layer rate that the path can transfer from source to end. In other words,*

the capacity of a path establishes an upper bound on the IPlayer throughput that a user can expect to get from that path.[36]

In a multiple linked path, the capacity from end to end can be defined as the minimal the capacity of the all the hops in the path.

$$C = \min_{i=1,\dots,H} C_i \quad (4)$$

where C_i is the capacity of the i -th hop, and H is the number of hops in the path.

3.5.2 Available Bandwidth

Another metric often used is the Available Bandwidth (ABW). The underlying propagation medium limits the capacity of a link. As a result, the available bandwidth is also affected by the traffic load that crosses the link, which often fluctuates over time. Therefore, refers to the underused portion of the link's entire capacity for a certain length of time. The available bandwidth on a hop over a certain time interval can be defined as followed: If C_i is the capacity of hop and μ_i is the average utilization of that hop in the given time interval, the average available bandwidth for that of hop is is described as the unutilized fraction of the capacity:

$$A_i = (1 - \mu_i)C_i \quad (5)$$

Once again, extending this definiton for a multi-path scenario, the available bandwidth of the end-to-end path is the minimum available bandwidth of all hops:

$$A_t = \min_{i=1,\dots,H} A_i \quad (6)$$

A “pipe model with fluid traffic” metaphor can be made to distinguish the available bandwidth and capacity. The total width of the pipe corresponds to the capacity of link and the unutilized area of the pipe describes the available bandwidth. The minimum capacity C_1 describes the link end-to-end capacity while A_3 describes the available bandwidth of the link. Both C_1 and A_3 represent the minimal measurements of each metric of the link, following the definition in equation 4 and 6.

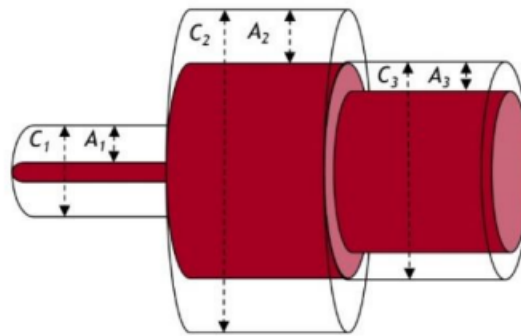


Figure 12: Pipe model with fluid traffic for 3-hop network [6]

3.5.3 TCP Throughput and Bulk Transfer Capacity

BTC, as described by Mathis and Allman [37], is the fastest a protocol that implements congestion control can forward packets from A to B. in TCP/IP networks, the Bulk Transfer Capacity is the maximum throughput obtainable by a single TCP connection. The connection must implement all TCP congestion control algorithms. The BTC can be simply defines as:

$$\frac{data_sent}{elapsed_time} \quad (7)$$

The assumption that all transport protocols should have comparable reactions to Internet congestion is central to the concept of bulk transport capacity. Indeed, the only kind of equity that is widely used on the Internet today is that the great majority of all traffic is carried by TCP implementations that share common congestion control methods, owing to a shared developmental background [37].

As it was suggested by Jacob Strauss and M. Frans Kaashoek [38]: In many applications, BTC is the one of the most valuable metrics to evaluate an Internet path. BTC is preferable to the capacity or available bandwidth estimation for any application that is utilizing the path and wants an estimate of what the network will support, or for choosing between paths. Nevertheless, there is a inherent extra load added to the network. BTC based tools inject, be default, the largest windows size possible in order to estimate the largest throughput possible. This phenomenon will inevitably affect the network performance.

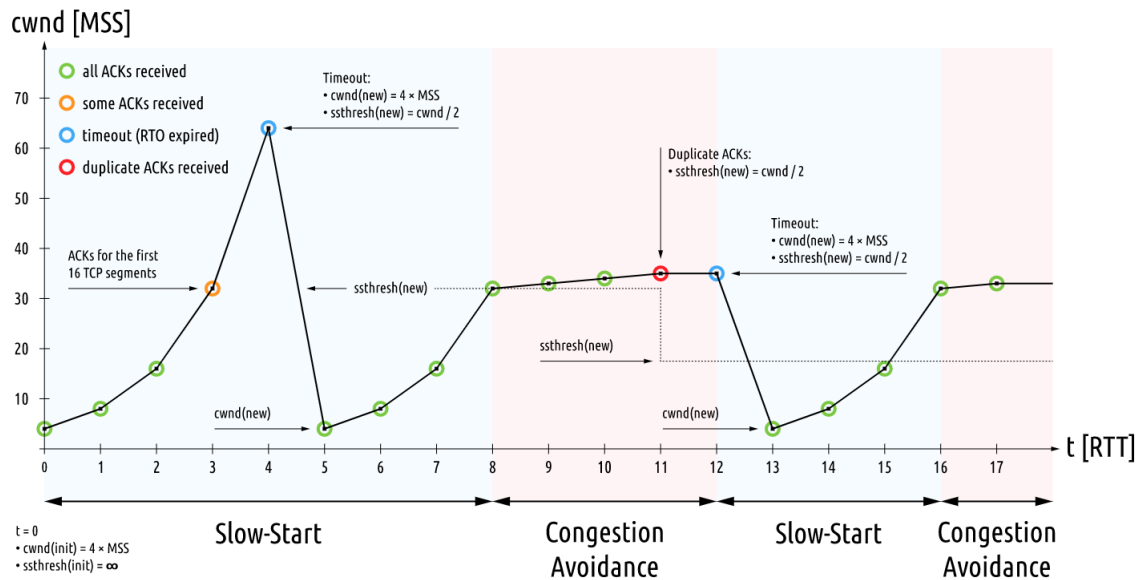


Figure 13: TCP (and therefore BTC) implements five standard congestion control algorithms

It is important to note that the available bandwidth and throughput are two fundamentally different. The main major difference is that the BTC is TCP specific. The Bulk Transfer Capacity depends on how TCP link shares bandwidth with other TCP flows, while the available bandwidth metric assumes that the average traffic load remains the same [36]. BTC depends heavily on network conditions. For example, buffer space on routers between A and B, queuing policies, and cross traffic on all hops will affect BTC [37].

3.6 Active Bandwidth Estimation vs Passive Bandwidth Estimation

The estimation of end-to-end bandwidth can be broken down into two main measurement methods, active and passive estimation. The fundamental difference between them is, that unlike passive measurement tools that are based on the non-intrusive monitoring of traffic, active tools are based on the concept of self-induced congestion.

3.6.1 Active Bandwidth Estimation

The active bandwidth estimation presents itself as the most simple way to measure bandwidth. The method's goal is to comprehend the network characteristics using artificially generated probe packets, sending them from the sender node to the recipient node. In addition to providing network operators with useful data on the characteristics and performance of the network, active bandwidth estimation tools enable end users, and consequently user applications, to independently perform network auditing, load balancing, server selection, and other tasks without needing access to network elements or administrative resources.[39]

These techniques can be further categorized into four sub-categories as briefly stated below:

- **Variable Packet Size (VPS) Probing techniques:**

VPS probing aims to measure the capacity of each hop along a path. The technique measures the Round-Trip-Time (RTT) from the source to each hop of the path as a function of the probing packet size. VPS sends multiple probing packets of a varying sizes from the sending host to each layer-3 device along the path. The RTT is calculated by the time a packet with the IPHeader "Time to Live" exceeds its time and returns a sends an error message to the source node corresponding to the internet control message protocol (ICMP) informing that the time was exceeded.

- **Packet Pair / Train Dispersion (PPTD):**

In this technique, the source sends two packets, a packet pair, of the same size, back and forth. The dispersion of a packet pair is then approximated as the time distance between the last bit of each packet.

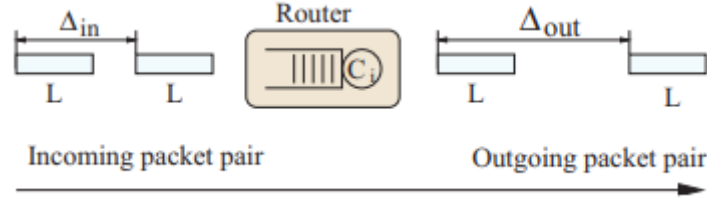


Figure 14: Packet Pair Dispersion as in [36]

If the dispersion prior to the path with capacity C_i is Δ_{in} , the dispersion after the link will be:

$$\Delta_{out} = \max(\Delta_{in}, \frac{L}{C_i}) \quad (8)$$

For a multiple link path, the dispersion at the receiver can be measured as:

$$\Delta_R = \max_{i=0,\dots,H}(\frac{L}{C_i}) = \frac{L}{\min_{i=0,\dots,H}(C_i)} = \frac{L}{C} \quad (9)$$

Where C is the capacity of the end-to-end path. The Bandwidth is calculated with mathematical formula that is derived by the sending and receiving gaps between the probing packets, thereby by simply measuring these gaps the ABW can be calculated as $\frac{L}{\Delta_R}$.

Packet Train extends the packet pair probing by adding multiple packets trains. The dispersion of a packet train at a link is the amount of time between the last bit of the first and last packets. With the Dispersion measure $\Delta_R(N)$ at the receiver is it possible to estimate the dispersion rate, also know as Asymptotic Dispersion Range (ADR), which relates to the utilization of all links in the path, which is equal to the end-to-end capacity:

$$D = ADR = \frac{(N-1)L}{\Delta_R(N)} \quad (10)$$

This technique has the inconvenience that it assumes that there isn't any cross traffic present in the path, which is an assumption that is far from realistic, specially in the video-streaming scenario.

- **Self-Loading Periodic Streams (SLoPS):**

This technique calculates end-to-end Available Bandwidth by monitoring probe packet's one way delay where probe packets of equal size (a "periodic stream") are sent to the path under consideration at a steady rate.

It is then monitored the bandwidth to the ABW so far. If the successive probe packet's one way delay increases that means that the rate is superior to the Available Bandwidth. The sender tries to adjust the speed of the transmission and probes the path while the receiver stores each periodic stream's one way delay information. The queue of the tight link will temporarily get overloaded if the stream rate exceeds the

path's available bandwidth. On the other-hand, if the stream rate is lower than the available bandwidth, the probing packets will go through the path without causing an increasing backlog at the tight link and their one way delays will not increase. The available bandwidth is estimated by leveling the maximum rate without generating overload.

- **Trains of Packet Pairs (TOPP):**

This approach uses the dispersion for gradually increasing rate of probe packets. TOPP repeatedly sends the train of packet pair at gradual rate between the source and the sink node. ABW is calculated as the maximum rate up to which the input sent from the source is smaller than the measured rate at the destination.

3.6.2 Passive Bandwidth Estimation

In the past years, research has been gradually focusing towards passive bandwidth estimation techniques to counter the limitations of active techniques. Active bandwidth estimation will, inevitably, always induce some additional overhead to the network traffic due to the presence of the probing packets. The idea of passive estimation is to observe traffic already present in the network and then estimate the bandwidth of the network based local consumption of bandwidth. Here the channel occupation is passively sensed to assess its busy time in proportion to its idle time. This idle-period is further used in conjunction with channel maximum capacity in order to evaluate the ABW of a node. In IEEE802.11, Medium Access Control, MAC, carrier-sensing range is utilized to sense the medium. Within the carrier-sensing range of a node, if any node is communicating to other node /nodes then the concerned node will sense the medium busy otherwise idle.

Passive methods use Probability Distribution Function (PDF) of packet inter-arrival in a TCP flow. The bandwidth is inferred by analysing variations in the PDF that describe characteristic behaviors. A spike is interpreted as a bottleneck with no substantial cross traffic, a spike bump as a low bandwidth bottleneck followed by a high bandwidth bottleneck, a train of spikes traversed bottleneck shared with a substantial amount of cross traffic and a train of spike bumps as a low bandwidth upstream bottleneck shared with a substantial amount of cross traffic[40].

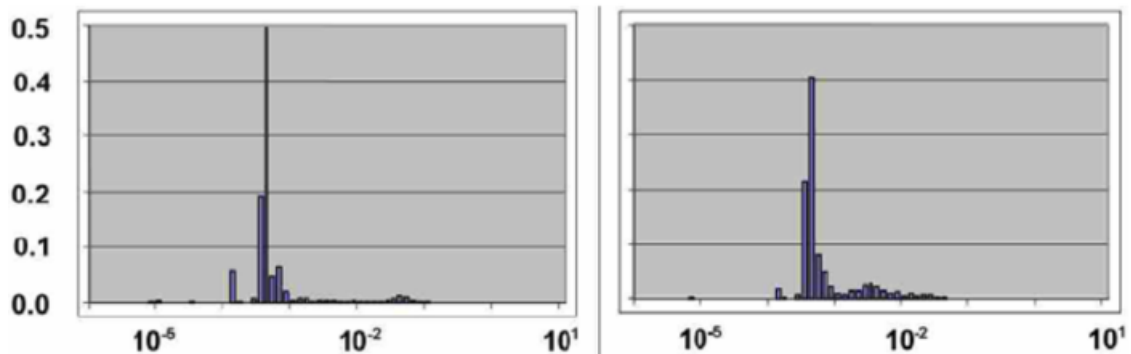


Figure 15: spikes in PDF of packet inter-arrival time indicate a small busy time interval as in [41]

3.6.3 Model-based Bandwidth Estimation

Model based techniques are based on the mathematical models. Since they can't anticipate the results after the introduction of additional flow in the network, passive approaches that are described in the preceding section are not very appropriate in predicting the ABW. They merely presume that when new flow network parameters are accepted, they will alter and have an impact on the actual ABW. For guaranteed and predictable service in packet networks, model-based tools allow the end-system to be able to manage and predict the performance of its active and extensible resources.

The model techniques are mostly deployed for multiple nodes in wireless packet networks and use a Markov Model to study the Distributed Coordination Function [43]. DCF is a protocol which uses carrier sensing along with a four way handshake to maximize the throughput while preventing packet collisions [42].

The Markov model is used to analyze the behavior of a single terminal and determine its "stationary probability" of sending packets over a broad time range. Based on the calculated probability, events that could happen within a generic time frame are studied in order to articulate the throughput.

3.6.4 Final Remark

Both the Passive and Model based techniques are still being used in very specific research scenarios, and consequently, the techniques are considerably harder to deploy, as there aren't many software tools available like in active estimation. They are described above to show that there are other methods for estimating bandwidth besides using active tools.

3.7 How Video Stream Quality Impacts Viewer Behavior

The biggest challenge for content providers is how ensuring a high-quality streaming experience for their viewers that also ensures uninterrupted reproduction, quick start-up and seamless quality transactions [44]. In the past years, Content Delivery Networks have provided the content creators with a way deliver higher quality video streams to a global audience [45]. CDNs also aim to track key metrics of viewer behavior that lead to better monetization. As such, a great deal of attention has been brought to the analysis of user engagement and view experience. Assessing the correlation between stream quality and viewer behavior is, unquestionably, one area of great interest. This also has profound implications for how a CDN, or a more broadly defined streaming service, must be designed.

Failures in video streaming are mostly inevitable. Nevertheless, different phenomenons on the video have different effects on the User Experience. Based on this premise, S. Shunmuga Krishnan and Ramesh K. Sitaraman, have studied the impact of video stream quality on viewer behavior using extensive traces from the Akamai's streaming network that include 23 million views from 6.7 million unique viewers [46]. The contributions of their study will be presented in a condensed manner in this section. Their work offers a great look at how users engage in their video experience and serves as a great basis for video Quality of Experience assessment.

In their study, Krishan and Sitaram aimed to establish a *correlational* link between a stream quality metric and viewer behavior metric and subsequently establish a *causal* link

between them.

They defined the metrics in two major categories explained in the following sections.

3.7.1 View-level Stream Quality Metrics

- **Failures:** Number of percentage of view that fail due to problems with the network, server or content. Failures address whether the stream was accessible to the viewer's initial viewing of of the video failed owing to a fault with the network, the server, or the material itself.
- **Average Bitrate:** Once the video begins to play, the average bit rate at which the video was reproduced on the viewer's screen serves a measure of the richness of the provided content.
- **Start-up Delay:** Total time in startup state.
- **Normalized Rebuffer Delay:** Total time the video stays in the rebuffer state.

3.7.2 Metrics for Viewer Behavior

- **Failures:** Number of percentage of view that fail due to problems with the network, server or content. Failures address whether the stream was accessible to the viewer's initial viewing of of the video failed owing to a fault with the network, the server, or the material itself.
- **Abandonment:** Viewer voluntarily decides to stop watching the video. Here, the metric is primarily concerned with abandonment where the viewer abandons the video even before it starts playing.
- **Engagement:** Total time in play state.
- **Return Rate :** Probability of user return within a time period.

3.7.3 Correlation vs Causation

It is important to note that certain external factors can have an impact on the metrics being investigated for causation. If one is attempting to find a causal relation between stream quality metric (say, startup delay) and a viewer behavior metric (say, abandonment rate), an external force may be causing the correlation but not necessarily the causation.

As an example, one can suppose that mobile users have less patience for videos to load because they are busy and "on the go," resulting in higher abandonment. Furthermore mobile users may experience longer startup times due to poor wireless connectivity. In this case, a correlation between startup delay and abandonment may not imply causation unless we account for the confounding variable of how the viewer is connected to the Internet.

Keeping in mind that extraneous forces may influence the perceived causal relation, in the particular cause of video streaming, some relevant external aspects have been taken into account by Krishan and Sitaram:

- **Content:** Both the quality perceived and viewer behavior may be influenced by the particular video being viewed. Some videos, for example, are more captivating than others, causing viewers to watch more of them. For example as put by the SSIMWAVE in their article titled "Is Live Sports Streaming Striking Out On Quality?": " Streaming live sports is one of the hardest video delivery tasks and sports fans are the least forgiving and the most vocal audience, especially if the video quality is not up to their expectations." [47]
- **Connection Type:** Both stream quality and viewer behavior may be impacted by a viewer's Internet connection method, including the device used and common connectivity traits.
- **Geography:** Viewer geography encompasses a variety of social, economic, religious, and cultural factors that can influence viewer behavior. Such a phenomenon could be significant in terms of how much stream quality influences viewer behavior. For example, social researchers have noticed that the threshold of patience that consumers exhibit toward a delay in receiving a product varies depending on the consumer's geographic location [48].

3.7.4 Statistical Power and Statistical Relevance

It is also important to note that when designing to design experiments that have "sufficient" statistical power to detect a "significant" effect, assuming such an effect does exist. It is also fundamental to evaluate whether the results are statistically significant or if they could have occurred by random chance. In their work Krishan and Sitaram, made the hypothesis testing by stating a null hypothesis that contradicts the assertion that they wished to establish. Afterwards, they evaluated the p-value of as the probability that they null hypothesis contradicts the assertion. Interestingly, except for one conclusion with a higher p-value that authors considered statistically insignificant, the results were extremely significant with p-values of $4 * 10^{-4}$ or less. This helped prove the robustness of their results in regards to their statistical significance.

The likelihood that the alternative hypothesis will be correct in its rejection of the null hypothesis is known as statistical power. The sample size, which is simply the number of matched pairs, hypothesis assert that a variable X has an effect of a certain magnitude on a variable Y, in the experiment; 1) the statistical significance level, and 3) the effect

size are three key factors that influence statistical power. Sufficient power is often defined as power that is at least 80%. The statistical power is largely affected by the size of the data. The effect size is defined as the difference between the percentage of matched pairs with positive outcomes and pairs with negative outcome, which, naturally, the lower, the better. The authors examined the minimum number of matched pairs required in a viewer abandonment experiment to detect a given effect size with statistical power at least 80%. They arrived at the conclusion that in order to achieve the necessary statistical power along with a low effect size, they required a very large data sample, which they assured via the Akamai's viewer behaviour traces.

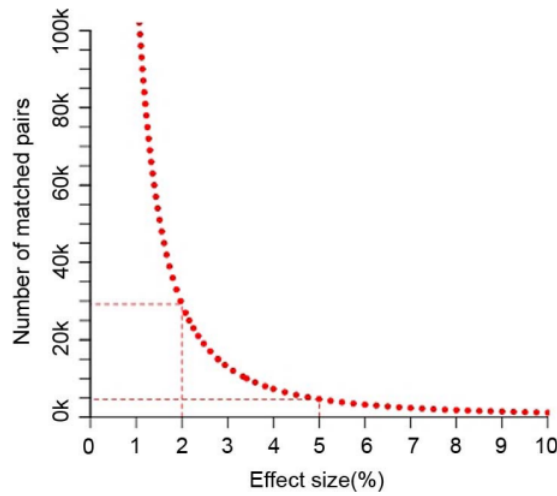


Figure 16: Minimum number of matched pairs required in a viewer abandonment experiment to detect a given effect size with statistical power at least 80% [46]

3.7.5 Their Findings

The work developed by Krishan and Sitaram have provided a great insight on how users respond to effects on the quality of their viewing experience, their conclusions are broken down in this section:

- **Viewer Abandonment**

To study this metric they came up with a function called `AbandonmentRate` that is defined as:

$$AbandonmentRate(x) = \frac{Impatient(x)}{Impatient(x) + Patient(x)} \quad (11)$$

here $Impatient(x)$ is all views that were abandoned after experiencing less than seconds of startup delay and $Patient(x)$ are views where the viewer waited at least time without abandoning. An $Impatient(x)$ translates to a view in which the user did not have the patience to wait for the time request for the video to startup. Based on this formula, they quickly inferred that : **An increase in startup delay causes more abandonment of viewers.** They also noticed that users tend to abandon at a higher rate for short videos than for long videos.

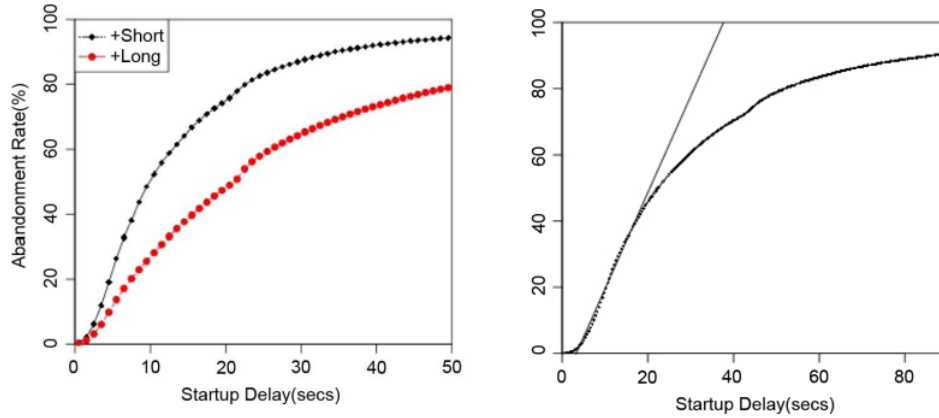


Figure 17: Viewers start to abandon the video if the startup delay exceeds about 2s. Viewers also tend to quit more quickly if the video is longer. [46]

They also observed how the user responded depending on their connection type that indicates how the corresponding viewer is connected to the Internet. Interestingly, they noticed that *viewers with better internet connection tended to abandon the videos at a much higher rates compared to viewers with slower connections*

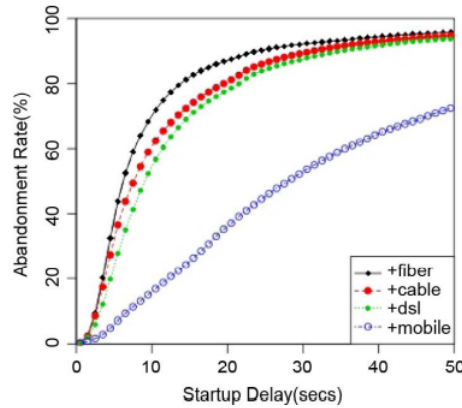


Figure 18: Viewers start tend to abandon the video much quicker if their internet connection is better. [46]

- Viewer Engagement** To comprehend viewer engagement, the authors studied the viewer play time per view. Firstly they realized that maintaining the user attention is a hard task. A noticeable fact is that a significant number of views have very small play time with the median play time only 35.4s. They also noticed that rebuffering is clearly correlated with less time watching the video. *The more interruption a video has, the worse will the perceived experience be for the user.* A viewer who experienced a rebuffer delay that equals or exceeds 1% of the video duration played 5.02% or less of the video in comparison to a similar viewer who experienced no rebuffering.

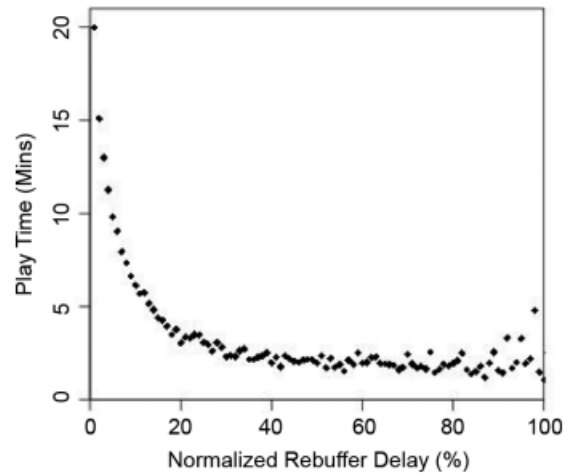


Figure 19: Correlation of normalized rebuffer delay with play time, the rebuffer delay was normalized by dividing for the video to rebuffer divided by the total video time. [46]

- Repeated Viewership** Once again, the authors found that failures in the video reproduction lead to a large reduction in the perceived Quality of Experience. The most of these, is if the video failed completely or if the user couldn't successfully visit the page. They concluded that *In comparison to a similar viewer who did not experience a failed video, a viewer who experienced a failed video is less likely to visit the website of the content provider again to watch additional videos within a given time frame.*

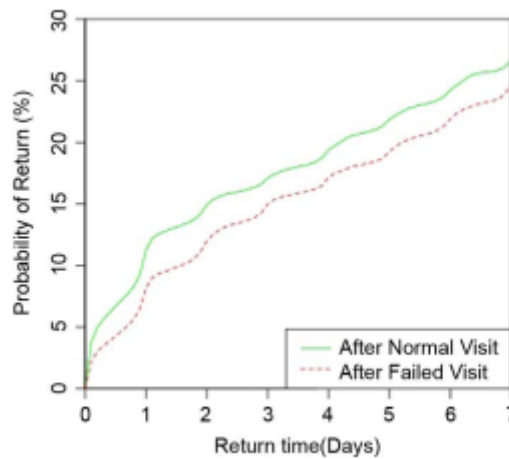


Figure 20: The probability of returning within a specified return time is distinctly smaller after a failed visit than after a normal one. [46]

3.8 Abrupt Bitrate Reduction vs Smooth Bitrate Reduction

Typically, the term bitrate and video quality are interchangeably used in video QoE studies. The larger the value of the bitrate, the more sharp the image will be, and the blocking phenomena will be less present. The way the bitrate quality is reduced may play a substantial role in the user Quality of Experience. Studies on this topic are few and usually refer to other aspects of "quality". Nevertheless some studies have concluded that gradual is much more preferred to viewers than abrupt quality reduction. In [49], Jim Harvey and Steven Le Moan investigate the ability of the human vision system to detect gradual changes in video quality, particularly chroma reduction. They concluded that the mean opinion score the users gave to the video with gradual quality reduction was almost the same value that they gave to the video playing with maximum quality. They also noticed that in scenes with more visual complexity, such as aerial view of a city, the viewers reported noticing less the quality reduction.

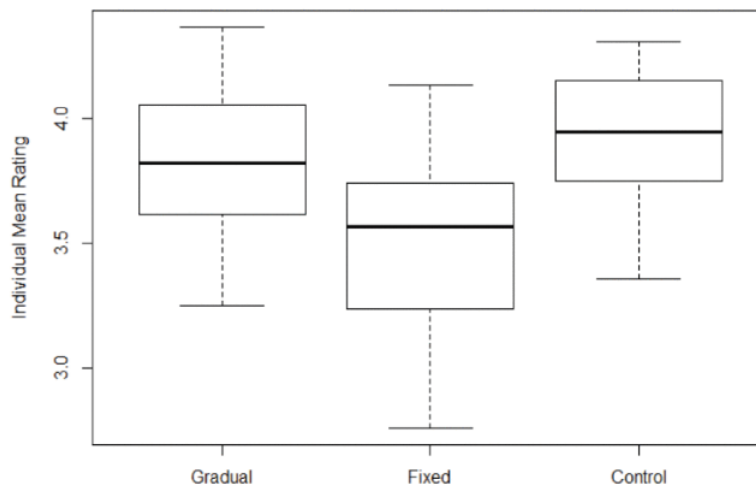


Figure 21: Individual Mean Opinion Score for the video with max quality is roughly the same than for the gradual quality reduction [46].

On the other hand, fixed reduction, brought a smaller Mean Opinion Score.

It stand to reason that the same effect may be observed on viewers for bitrate reduction and as such, it's a factor worth considering while adjusting the SmoothMV's video bitrates.

3.9 Perception of Quality and Video Content

According to research conducted in the field of video quality evaluation, the correlation between video bitrate and perceptual quality is non-linear [51] [52]. Furthermore, a video with high motion, for example, a video of a basketball game, and a video with low motion, for example, a video of a person talking in a relaxed environment, have distinct characteristics that produce distinctive interpreted qualities. High motion scenes will have higher values of spatial frequency and losing quality in these will have a bigger impact on perceived quality.

Fig. 21 exemplifies the the non-linear relationship between bitrate and the Structural SIMilarity plus (SSIMplus) [53] perceptual quality.

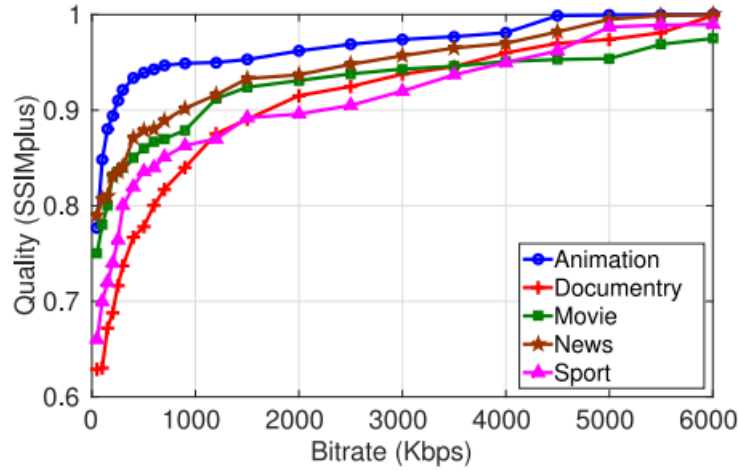


Figure 22: Illustration of quality versus bitrate trade-off as in [54].

The effect has been observed to be present even in intra-stream segments. The perceived quality may be lower even for the same allocated bitrate.

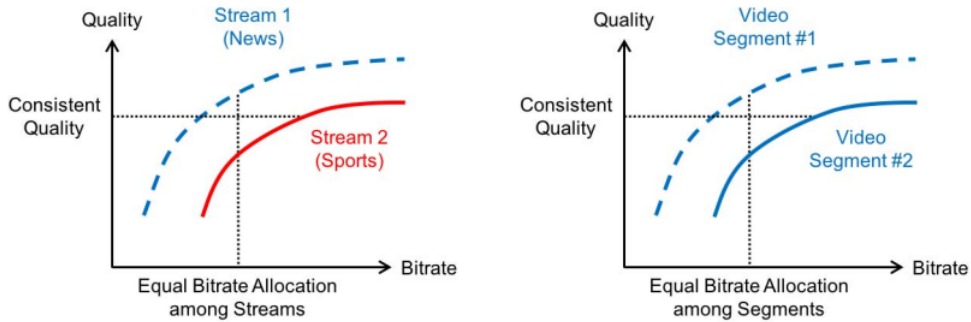


Figure 23: At the same encoding bitrate, varying inter-stream (on the left) and intra-stream (on the right) scene complexity results in varying display qualities, or vice versa as in [54]

3.10 Related Works

3.10.1 Adaptive Bitrate Streaming

ABR is a type of algorithm in which the video player dynamically adjusts a stream's compression level and video quality to match bandwidth availability. ABR ensures that the video streaming maintains constant, adapting the bitrate according to the bandwidth measurements. Each protocol, has its own ABR algorithm, to decide which bitrates to

download next. Different algorithms have been developed based on different techniques to ensure continuous streaming. To decide which bitrate to choose next, throughput-based also named client-based algorithms track the speed at which earlier video chunks were downloaded. Buffer-based algorithms attempt to control buffer occupancy in order to ensure that enough video is available for playback. If the media in the local buffer runs out, the next bitrate will be lower to keep up with playback. Some ABR algorithms make a Network-assisted adaptation, taking into account explicit information from within the network, and some make a Hybrid adaptation, using information from any combination of the client, server(s), and network.

In the following section, some of examples of ABR algorithms are introduced based on the entity of the system where the logic is implemented.

3.10.2 Client Based Bitrate Adaptation

Client Based algorithms are the most typically used for bitrate adaptation. These schemes attempt to adapt to bandwidth variations by selecting the appropriate video bitrate based on multiple metrics such as available bandwidth, playback buffer size, and so on. Most algorithms make representation decisions based on the measured available network bandwidth, which is usually calculated as the size of the fetched segment(s) divided by the transfer time - throughput-based.

- **Llama: Low Latency Adaptive Media Algorithm**

Llama is a client based novel ABR algorithm targeted for live streaming scenarios. Live streaming requires networks with low latency, which in turn will have smaller buffer sizes. Llama is developed specifically for small buffer live streaming scenarios [50].

Llama estimated the available bandwidth by observing the variations in the throughput of video chunks received in the client size. Large variations in throughput may occur due to changes in the network conditions. In order to make a more stable estimation of the bandwidth, Llama calculates a moving harmonic average. The algorithm then compares the last throughput measurements with the current segment's bitrate received and changes the playback's bitrate accordingly (increases the bitrate if the bandwidth is larger, decreases the bitrate if the bandwidth is lower).

Playback Buffer-Based Adaptation

In this type of adaptation, the algorithm decides the next segment's bitrate using video playback based on the playout buffer occupancy. These techniques measure the duration of video left in the buffer, and accordingly download a higher bitrate segment (if the buffer is sufficiently occupied) or a lower bitrate segment (if the buffer is running low).

- **Buffer Occupancy based Lyapunov Algorithm (BOLA)**

BOLA is the buffer-based algorithm that is implemented in the dash.js player [55]. The algorithm is an online control algorithm that treats bitrate adaptation as a utility maximization problem. The technique aims to maximize the playback utility and playback smoothness of the streaming protocol by using Lyapunov optimization. As the network's throughput varies, the objective is to avoid stalls caused by an empty buffer while selecting high-quality encoding. the utility is associated with the average bitrate and rebuffering time, while adapting to network changes to account for better QoE.

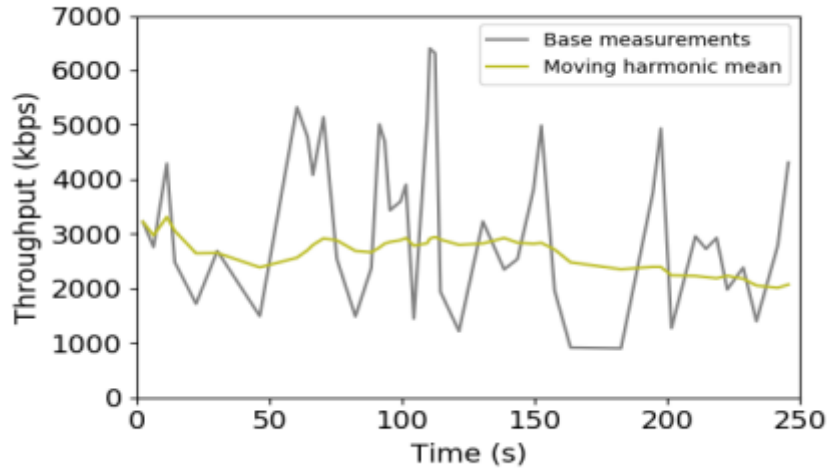


Figure 24: Llama Sample Trace with Base Measurements and Moving Harmonic Mean Plotted [60]

3.11 Server-Based Adaptation

Server-based systems do not require client cooperation and utilize a bitrate handling method at the server side. The bitrate shaper therefore implicitly controls the switching between bitrates. The client still makes its own decisions, but those decisions are largely determined by the server's shaping method. These algorithms usually deploy traffic shapers for streaming in the presence of multiple HTTP adaptive streaming players competing for the available bandwidth to tackle instability and unfairness issues.

Detti et al. [56] proposed a tracker-assisted adaptation strategy in the presence of network caches. The proposed architecture essentially involves clients communicating with the server through a common proxy and an extra server with tracker functionality that controls the clients' statuses and enables them to share information regarding their statuses.

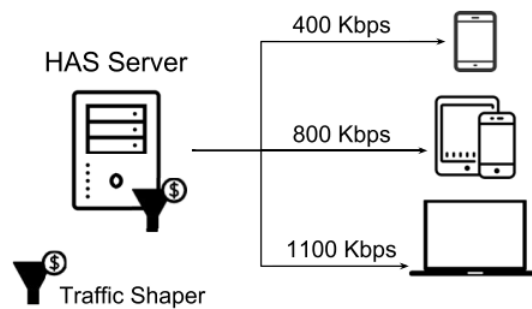


Figure 25: Basic Server-based Bitrate Adaptation [54]

The complexity and overhead of server-based bitrate adaptation schemes are substantial, notably as the number of clients rises.

3.12 Network-Based Adaptation

Network based bitrate adaption algorithms allow the HTTP adaptive streaming players to pick the preferable bitrate based on the network measurements. This is accomplished by collecting measurements about network conditions and informing clients about the best bitrates to download. This is done by a agent/proxy deployed in the network to monitor the network status and conditions. The external agent offers network-level information that allows the clients to efficiently make use of the resources the network provides.

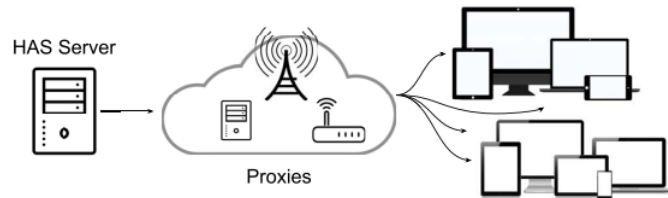


Figure 26: Basic Network-based Bitrate Adaptation [54]

3.12.1 QDASH - QoE-aware DASH

QDASH [57] is a network based bitrate adaption algorithm that deploys two proxy modules between streaming server and the client that aims to avoid video oscillations by ensuring a gradual change in bitrate levels through the use of integrated intermediate levels. QDASH utilizes the QDASH-ABW Probing Methodology to estimate the bandwidth. Unlike typical bandwidth estimation tools which use customized probing packets, QDASH-ABW uses inline measurements that employ the media data packets directly to determine whether a certain sending rate is supported by the current available bandwidth. This prevents the common problem involving the TCP based bandwidth estimation tools, which is their intrusiveness. Moreover, an additional TCP connection may not be accurate because the new TCP connection may not go through the same network path as the media data because of the load balancing.

QDASH-ABW works by sending a packet-train at the same time and at the same rate as the packets being sent by the server. The packets are then intercepted by the proxy. The proxy estimates the bandwidth and sends the packets to the client at a rate adjusted to the network conditions available.

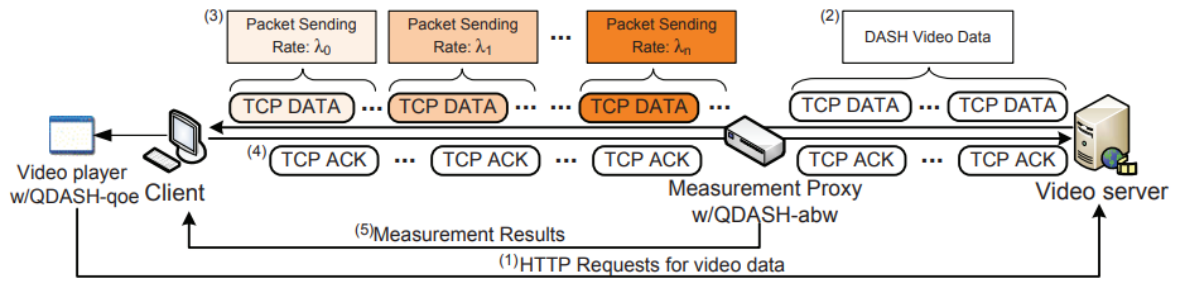


Figure 27: The overall QDASH system's architecture [57]

Chapter 4

4 Methodology

In the following section, a detailed description of the work developed in this thesis will be presented. The details of the development of the system and subsequent obstacles and solution encountered will be present sequentially as the work was developed. This chapter begins by providing a brief explanation about the existing SmoothMV and how the solution developed was integrated in it.

4.1 System Breakdown

SmoothMV's architecture, as presented in previous chapter, is composed by two sides, the server core and the client core. The server core focuses on encoding/post-processing tasks, real-time handling of user requests and sending the appropriate versions of available content. The Client Core performs end-user tasks to ensure smooth content playback, head tracking, view identification and switching, and QoE assessment. This breakdown of the system's architecture approaches SmoothMV in a functional point of view. However, in more a more technical point of view, the system's client core is divided in three modules: The View Selection Tracking module and the View Buffering Module and the Server.

4.1.1 View Selection Module

The View Selection Module is based on the Intel Realsense Software Development Kit's Face Tracking Kit as illustrated in figure 28. The module includes a client with an application panel where the blue dot corresponds to the user's head tracking coordinates. The coordinates are inserted in the Hot&Cold matrix. As mentioned in the preceding section, the matrix uses the user's center of attention to transition to a new view, and buffering other views, depending on their position within the panel's grids and semigrids.

Besides handling the head tracking and the view selection, the View Selection Module also handles the quality selection corresponding to the views.

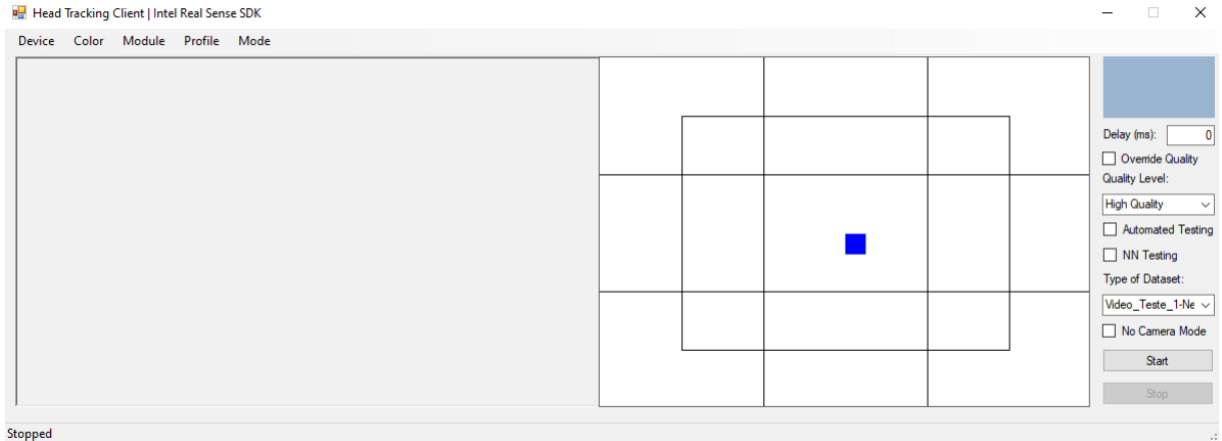


Figure 28: Client Side Head Tracking Client

The client also incorporates some options related to testing the neural network and choosing the Recurrent Attention Mode dataset corresponding to the desired video.

The View Selection Tracking Module saves the all parameters gathered in a log file. Every time the media is played, the SocketData function is called. Here, a .txt file is generated, whose name is of the format "log-file-client-yyyy-MM-dd-HH-mm-ss", according to the time and date the function was called. On this file, around every 0.05 seconds (at about 20 frames per second, or lower according to processor speed), a log file is written as seen in Figure 29.

```
Head Tracking - Log Data
Grid Position: 0.0
SemiGrid Position: 0.0
Pitch: 0.0
Yaw: 0.0
Roll: 0.0
X Value: 228
Y Value: 159
Face Landmarks:
Automated Quality Setting: 0
Quality Level: 1

Head Tracking - Log Data
Grid Position: 0.0
SemiGrid Position: 0.0
Pitch: 0.0
Yaw: 0.0
Roll: 0.0
X Value: 228
Y Value: 159
Face Landmarks:
Automated Quality Setting: 0
Quality Level: 1

Head Tracking - Log Data
Grid Position: 0.0
SemiGrid Position: 0.0
Pitch: 0.0
Yaw: 0.0
Roll: 0.0
X Value: 228
Y Value: 159
Face Landmarks:
Automated Quality Setting: 0
Quality Level: 2
```

Figure 29: Log File generated with three instances of logging

The information present in the log file is sent through a socket to the View Buffering Module as it is written.

4.1.2 View Buffering Module

The View Buffering Module is responsible for downloading the segments and reproducing the video. The module collects head tracking data from the View Selection Model via socket and assigns the views accordingly. The views are chosen and pre-buffered based on the x and y values.

The module is divided in 6 parallel threads, each one of them responsible for handling different parts of the system: *the network thread*, operates the segments selection, manipulation and sequential download, *the tracking thread*, responsible for binding with the View Selection Module and receiving and storing the head tracking information, *QoSMetrics*, handles the quality transactions and calculates network and throughput, and *playerEmulation*, which takes care of the view reproduction.

Initially N number of segments are downloaded and stored in a initial buffer. The N segments are defined in the variable NUMBER_OF_SEGMENT_BUFFER. After the video starts playing, based on playback timing and position received within the Hot&Cold matrix, M number of segments are sequentially downloaded: from a single view if in the Inactive stage, or from additional views if in the Buffering stage. The segments are firstly stored in a temporary buffer of size M defined in the variable BUFFERING_PROPORTION.

```

ViewBufferingModule_Gustavo
100 440k 100 440k 0 0 21.5M 0 --:--:-- --:--:-- --:--:-- 21.5M
Downloading Segment 34 from Vista 1 on Queue 2 at Time 21.880741
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 484k 100 484k 0 0 21.4M 0 --:--:-- --:--:-- --:--:-- 21.4M
Downloading Segment 35 from Vista 1 on Queue 2 at Time 22.505905
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 434k 100 434k 0 0 14.1M 0 --:--:-- --:--:-- --:--:-- 14.6M
Downloading Segment 36 from Vista 1 on Queue 2 at Time 23.131069
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 476k 100 476k 0 0 23.2M 0 --:--:-- --:--:-- --:--:-- 23.2M
Downloading Segment 37 from Vista 1 on Queue 2 at Time 23.756233
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 428k 100 428k 0 0 19.0M 0 --:--:-- --:--:-- --:--:-- 19.0M
Downloading Segment 38 from Vista 1 on Queue 2 at Time 24.381397
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 460k 100 460k 0 0 19.5M 0 --:--:-- --:--:-- --:--:-- 19.5M
Downloading Segment 39 from Vista 1 on Queue 2 at Time 25.006561
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 418k 100 418k 0 0 16.3M 0 --:--:-- --:--:-- --:--:-- 16.3M
Downloading Segment 40 from Vista 1 on Queue 2 at Time 25.631725
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:--

```

Figure 30: Video Buffering Module Downloading the Initial Segments

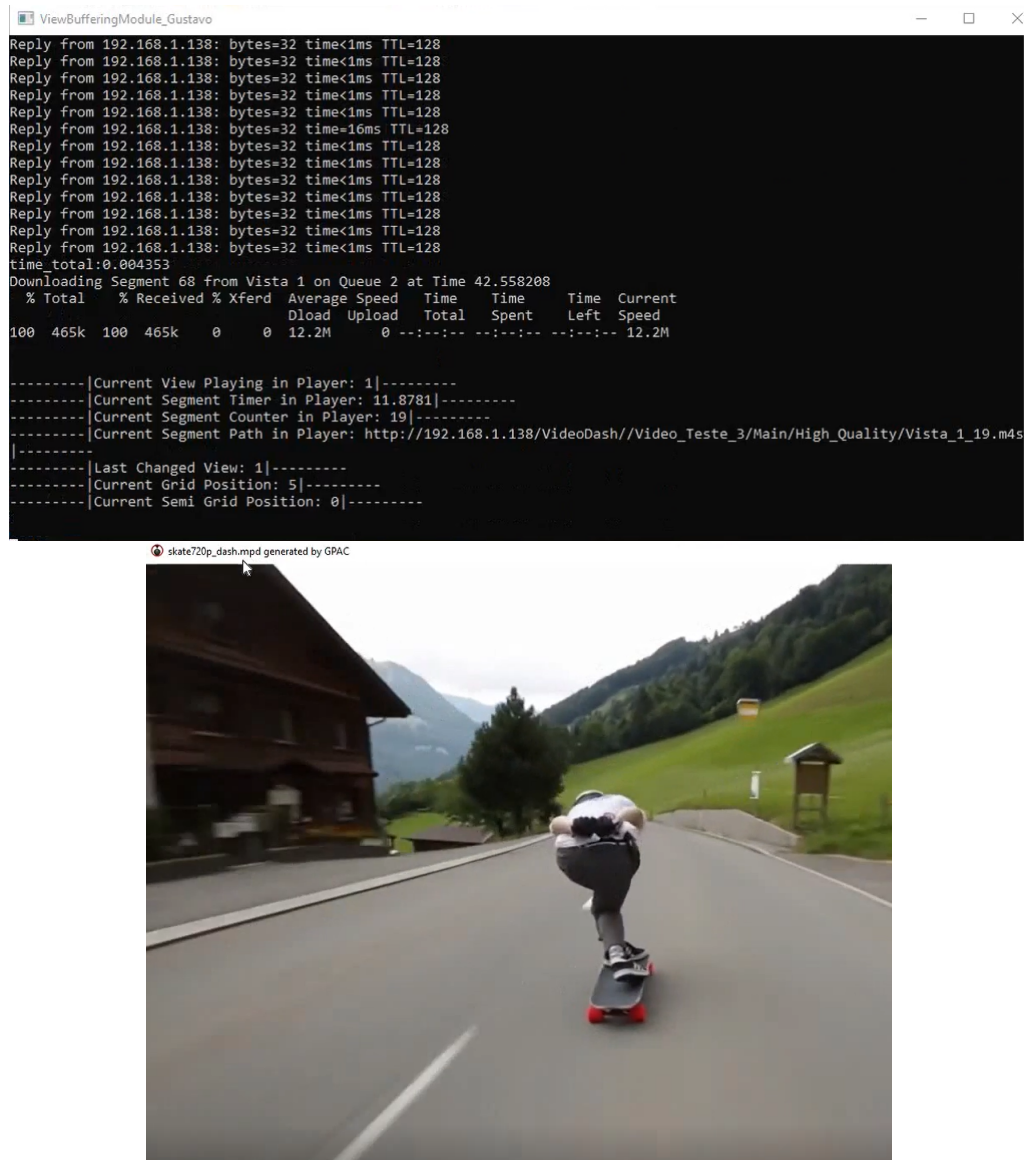


Figure 31: Video Buffering Module Downloading the Segments and sequentially playing them

4.1.3 Server Core

the Server core, has presented in the context section, is divided into three functional layers, The **Media Handling Layer** which is responsible for encoding multiple variations of the raw multi-view content and for preparing the content in conformance to MPEG-DASH using the FFMPEG and GPAC software packages; **the Request Handling layer** which manages HTTP requests and the **the View Streaming Layer** which uploads the content to the client. The server stores the videos in a folder with a subdirectory for each bitrate. Each subfolder contains the video encoded for all the views, a mpd file and a descriptor file named View Model that is responsible for identifying the different perspectives of the multi-view content.

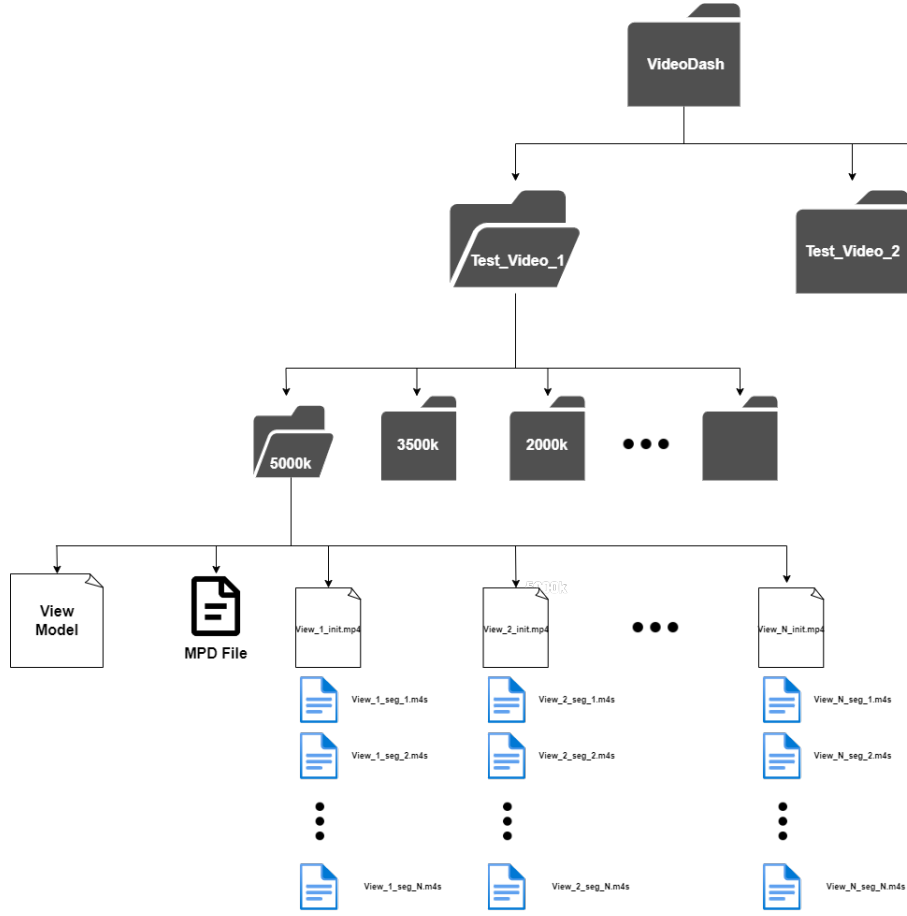


Figure 32: Server Core Folder Organization

The View Selection Module makes the GET requests to the server which in turn responds to the Client with minimal delay, using an Apache HTTP server. The Apache Server is deployed using the Wamp software [58]. In order to make the requests to this server, it is only required to be on same LAN (or have the Wamp Server publicly open) and make the curls requests to the server's IPaddress following /VideoDash/-Main/Video.Test_N/5000K/MPD File.

The overall system can be summarized in the following scheme:

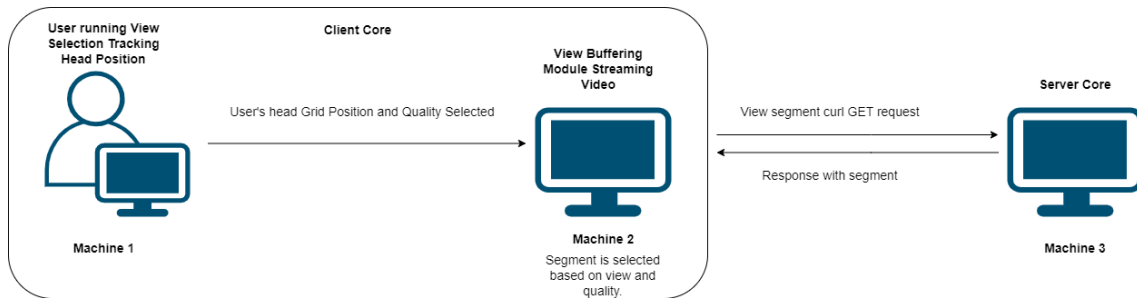


Figure 33: Overall SmoothMV Scheme

4.2 System Developed for the Bandwidth Estimation

4.2.1 SmoothMV Scheme Adaptation

To better suit the work that developed in this thesis, the SmoothMV system scheme was adapted. The modifications kept every SmoothMV's operation intact.

As the server's only function is storing the videos and sending the segments to the client, an adaptation was made too more comfortably suit the system to be developed: The scheme was readjusted to accommodate two machines, Machine 2 runs the View Selection Module and serves as the server and Machine 1 runs the View Buffering Module and reproduces the video.

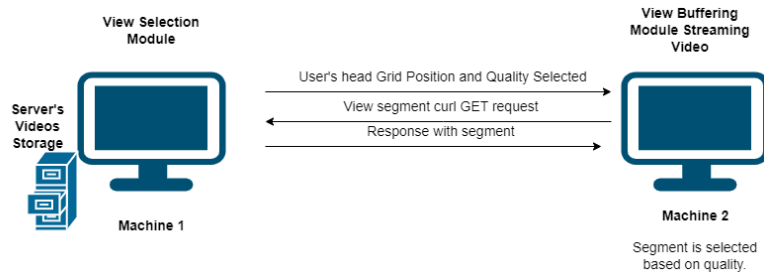


Figure 34: Adapted SmoothMV Scheme

The workbench required for this work was provided by INESC-TEC. To develop the work necessary and to conduct the experiments, the student was provided with a desk, a computer running Windows 10 and a router. The computer supplied by INESC-TEC served as Machine 1, running the View Selection Module and storing the videos, and the student's personal laptop served as Machine 2, running the View Buffering Module and streaming the videos. All the work was done in an Ethernet connected environment.

4.2.2 Adaptive Bitrate Streaming (ABR) Scheme

It should be noted that SmoothMV, in its default form, already presented a bitrate adaptation functionality in the View Buffering Module. Under this version, bandwidth was equated to the TCP Throughput attained at the optimal TCP Receive Window Size, much like the client-based ABR algorithms presented in the previous chapter. However, this proved to be a somewhat suboptimal approach. For this thesis, a ABR algorithm was developed as a Network-Based bitrate adaption: the bandwidth was estimated based on network measurements.

The system designed was built to accept the various tools in order to conduct a thorough analysis of the optimal bandwidth measurement tool. The conceived design is based on the typical Network-Based ABR scheme: a proxy system is deployed in the network to monitor the network status and conditions. The external agent offers network-level information that allows the clients to efficiently make use of the resources the network provides.

The vast majority of bandwidth estimation tools are deployed in a similar fashion. There are two hosts and the purpose is to measure the available bandwidth from one machine to the other. This is done by running the software in both ends, in one end as the server, and the other end as the client. The server opens a port and waits for the communication to start and the client starts sending the packets to the server's IP. The bandwidth measurements are then presented on the server side, the client side or both sides, depending on the tool.

In the system's scenario the segments are sent from the server, Machine 1, to the View Buffering Module to play the video, Machine 2. As such, the bandwidth estimation packets much sent in the same direction, so Machine 2 works as bandwidth measurements server, waits for packets to be sent, and machine works as the bandwidth measurements client, sends the probing packets. The scheme can be visualized in Figure 35:

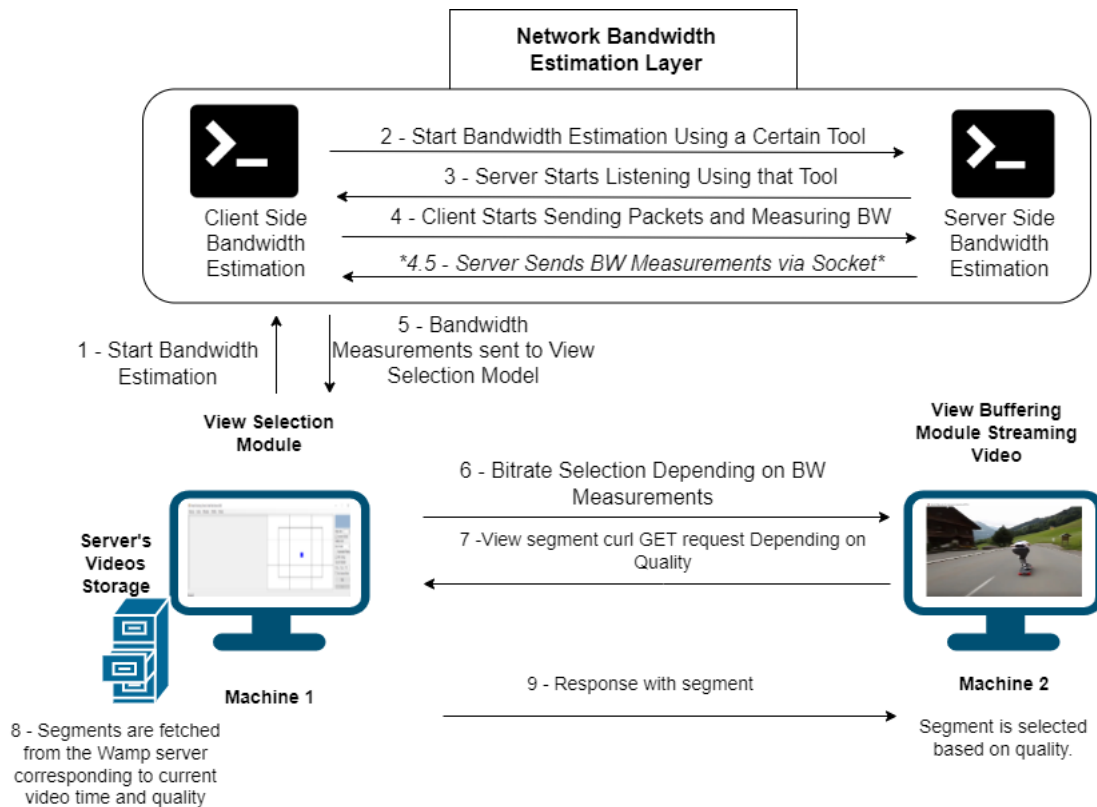


Figure 35: Adapted SmoothMV Scheme

The new Network Bandwidth Estimation Layer works in the following manner:

- **1 - Start Bandwidth Estimation** - In the View Selection Module, the user selects the option *Override Quality* and *No Move Mode* to start the process. The adaptations made to the View Selection Model and Software Developments aspects of the new process will be explained in the next section;
- **2 - Start Bandwidth Estimation Using a Certain Tool** - The user picks a certain tool and a socket message is sent to the server informing to start the listening process for that particular tool;

Machine Specifications					
Machine 1		Machine 2		Network Specifications	
Windows 10 Pro 22H2		Windows Home 21H2		Gigabit Wired Ethernet	
Intel(R) Core(TM) i7-6700U CPU @ 3.40GHz 3.41 GHz		Intel(R) Core(TM) i7-6500U CPU @ 2.50GHz 2.60 GHz		Router TP-Link AC1200 Archer C1200 Dual-Band Gigabit	
16GB RAM		8,00 GB RAM			

Table 1: Machines and Network Specifications

- **3 - Server Starts Listening Using That Tool** - The server informs the client that it's listening and it can begin sending the packets;
- **4 - Client send probing packets to the Server and estimates Bandwidth (BW)** - The client starts sending the packets and it calculates the bandwidth. The way the calculations are made depends on the tool defined;
- **4.5 - Server sends BW Measurements** - Some of the tools don't present the bandwidth estimations on the client but present them on the server side. As it is in the client's side the process to pick the quality depending on the BW, it is necessary to send this information via socket;
- **5 - BW Measurements send to View Selection Model** - In this phase, the Network Bandwidth Estimation Layer's Client sends its measurements to be interpreted by the View Selection Model;
- **6 - Bitrate /Quality Selection Depending on BW Measurements** - The View Selection model selects the quality / bitrate based on the BW values and communicates it to the Video Buffering Module. The way this is done will be explained ahead in this document;
- **7 - View Segment curl GET Request Depending on Quality** - The View Buffering receives the information about the selected quality and sends HTTP GET requests to the Wamp Server to sequentially download the segments.
- **8 - Segments are Fetched from the Server** - Based on the current video play time and the quality selected, the segments are fetched;
- **9 - Response from Segments** - Finally the View Buffering Module receives a new segments and plays it in the video.

4.3 Software Development

The system's software development will be discussed in a non-exhaustive manner in the next section. Considering discussing code development is a tricky task, only the most

significant parts will be covered succinctly. All code was written in C#. In order to start the process of measuring the bandwidth, the user has to simultaneously select the check the options *Override Quality* and *No Move Mode*. No Move Mode was added for the sake of simplicity, as there was no need for view switching for the work that was produce and selecting this causes the system to start while remaining in the central view. After this is done and start button is pressed, the software begins running the bandwidth measurements and communicating with the View Buffering Module. The client-side bandwidth estimation was developed directly in a new library within the View Buffering Module code.

All the tools are shell-based programs. To begin measurements, commands are entered into a terminal on both machines. In order to deploy the tools, a new parallel thread for the measurements was introduced in the View Selection Model. Once the new thread started, a socket message is sent to the server informing it to start listening for that particular tool and a new process is started that opens a terminal in the folder where the tool is located. Once the terminal is open, a command is entered to begin sending probing packets to the server's IP.

Iperf [59] was the first tool used to make bandwidth measurements. Iperf estimates bandwidth by measuring the largest achievable TCP Throughput. As Iperf has a Windows Operating System version, it is only necessary to start a process to open a command prompt in the Iperf's folder with the commands to start sending packets to the server's address. The process waits for the measuring to be complete and then returns the values in text form in the terminal. The values are then converted from string to float and kept in a list. The user is then shown the recorded data, and the values obtained are saved in a log file. A visual Windows Form was included in order to clearly display the findings to the user.

EstimationNo	EstimationValue
1	354 Mbytes/s
2	352 Mbytes/s
3	346 Mbytes/s
4	348 Mbytes/s
5	341 Mbytes/s
6	345 Mbytes/s

(a) Windows Form

```

Estimation number: 0/ Estimated bandwidth -> 248.178 Mbytes/s
Running Average: 82,726
Estimation number: 1/ Estimated bandwidth -> 174.583 Mbytes/s
Running Average: 140,920333333333
Estimation number: 2/ Estimated bandwidth -> 180.576 Mbytes/s
Running Average: 201,112333333333
Estimation number: 3/ Estimated bandwidth -> 191.245 Mbytes/s
Running Average: 182,134666666667
Estimation number: 4/ Estimated bandwidth -> 180.352 Mbytes/s
Running Average: 184,057666666667
Estimation number: 5/ Estimated bandwidth -> 169.362 Mbytes/s
Running Average: 180,319666666667
Estimation number: 6/ Estimated bandwidth -> 183.24 Mbytes/s
Running Average: 177,651333333333
Estimation number: 7/ Estimated bandwidth -> 168.039 Mbytes/s
Running Average: 173,547
Estimation number: 8/ Estimated bandwidth -> 165.764 Mbytes/s
Running Average: 172,347666666667
Estimation number: 9/ Estimated bandwidth -> 163.986 Mbytes/s
Running Average: 165,929666666667
Estimation number: 10/ Estimated bandwidth -> 173.159 Mbytes/s
Running Average: 167,636333333333

```

(b) Log File

Figure 36: Windows Form designed to show the current Bandwidth Estimation and corresponding Video Quality and Log File generated by the Bandwidth Estimation Layer.

As for the server side of the new Bandwidth Estimation Layer, a program was made that waits for the socket connection to start listening for packets. This program is launched when the View Buffering Module is started. After the request is received, the server starts listening for packets corresponding to the tool specified in the request.

As extra drop-down list was added to the Head-Tracking's GUI in the View Selection Module so the user can pick the tool he/she wishes to use. The tools chose will be explained later.

The GUI also has a drop-down list with the different available qualities, which, based on the user's selection, will assign to the variable `QUALITY_LEVEL` a value from 0 to 4 depending on the quality, 0 being the highest quality/bitrate, 5 being the lowest quality/bitrate. The original View Selection Module had 3 quality level, Low Quality, Medium Quality and High Quality but two more qualities were added: Very High Quality and Very Low Quality.

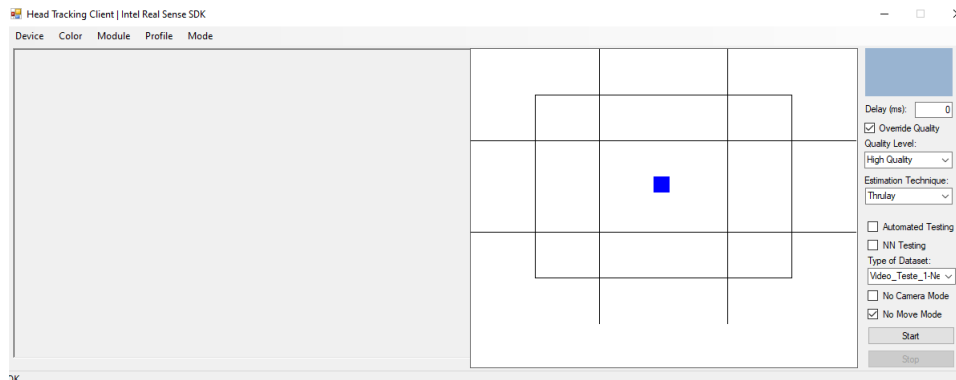


Figure 37: Updated Head Tracking GUI.

4.3.1 Deploying the Bandwidth Estimation Tools

Iperf was the first tool to be deployed and, as it was previously mentioned, has an available Windows's Operation System version, which made the its deployment relatively straightforward. SmoothMV is completely designed in Windows and, as such, it is completely bound to the its networking technologies and stack. This posed as a real problem for testing the tools as *most bandwidth estimation tools are designed to be used in a Unix-based Operating System* and as consequently can't be deployed in a Windows Command Prompt.

The inability to deploy the tools proved itself to be a somewhat complicated problem to solve. Nonetheless, a variety of workarounds were explored. The first of these was using a virtual machine running Linux Operating System (OS) in each computer. Most Virtualization softwares allow Bridged Networking which is used "for more advanced networking needs, such as network simulations and running servers in a guest. When enabled, Oracle VM VirtualBox connects to one of your installed network cards and exchanges network packets directly, circumventing your host operating system's network stack" [60]. The idea was making the bandwidth estimations exclusively in the Virtual Machines and then sending the values to the View Selection Module via socket communications.

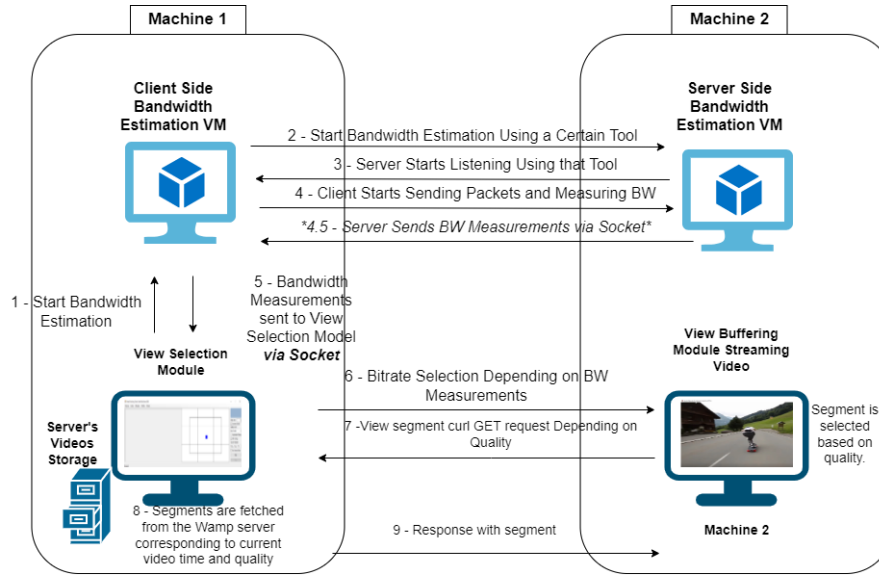


Figure 38: Adapted SmoothMV Scheme with Virtual Machines

Due to the increased resource utilization and additional overhead, this rapidly turned out to be a subpar option.

The best solution was eventually discovered, which was to use the Windows Subsystem for Linux (WSL). WSL requires fewer resources (CPU, memory, and storage) than a full virtual machine. WSL allows running Linux command-line tools and apps alongside Windows command-line and desktop apps. It is also possible to access Windows files from within a Linux's terminal environment. As a result, the same collection of files may be employed by Linux command-line tools and Windows applications [61].

```
gustavo@DESKTOP-HSNFCH2: /mnt/c/Users/pelay/Desktop/tools/pathrate-master
gustavo@DESKTOP-HSNFCH2:/mnt/c/Users/pelay/Desktop/tools/pathrate-master$ ls
CHANGELOG  confcache  config.status  makefile  pathrate_rcv  pathrate_rcv_func.c  pathrate_snd.h
COPYING    config.guess  config.sub  makefile.in  pathrate_rcv.c  pathrate_snd  pathrate_snd_func.c
README     config.log  configure  pathrate.h  pathrate_rcv.h  pathrate_snd.c
gustavo@DESKTOP-HSNFCH2:/mnt/c/Users/pelay/Desktop/tools/pathrate-master$ ./pathrate_rcv -s 192.168.1.121
pathrate run from 192.168.1.121 to DESKTOP-HSNFCH2 on Mon Jan 16 12:48:00 2023
```

Figure 39: WSL running Bandwidth Estimation Tool in Ubuntu's Bash shell

With Windows Subsystem Linux installed, it became possible to use the Bandwidth Estimation Tools in the Windows OS without blocking any of networking functionalities as it previously happened. Moreover, installing WSL with Ubuntu's terminal environment permits calling the Linux terminal in a C# Windows Application, which served the system's purpose perfectly.

4.4 Bandwidth Estimation Tools

In the section that follows, the tools used are briefly exposed and explained. To assess the bandwidth as thoroughly as possible, tools with various techniques were put into place. Some assess the network's capacity, others the available bandwidth, while others approximation the bandwidth as the highest throughput possible. The tools were evaluated and compared in regards to their *accuracy, robustness (stability, response time, response to interference) and applicability to the scenario at hand* in the Results Section.

A lot of different techniques were experimented, with varying degrees of success. Unfortunately, some of these tools are very outdated and as such the results can be very unpredictable. The majority of them were released in the 1990s and early 2000s (except some Available Bandwidth and Bulk Transfer Capacity Tools), which helps justifying their instability when applied to very high speed networks.

It should be noted that *a lot of the tools documented in the literature are no longer available or simply don't work with contemporary C compilers*. Naturally, the availability of the tool had a significant impact on the tool selection. Nevertheless, in this thesis, an effort was made to vary the tools in order to evaluate the most frequently utilized methods.

4.4.1 Iperf

Iperf is a tool for measuring the theoretical maximum bandwidth on IP networks. It allows users to fine-tune various timing, buffer, and protocol parameters (TCP, User Datagram Protocol (UDP), Stream Control Transmission Protocol (SCTP) with IPv4 and IPv6). Iperf allows the user to configure multiple parameters that can be used for network testing, optimization or tuning like the windows size, number of parallel streams, reporting interval, Number of bytes to transmit, testing time, etc. However, Iperf, by default, tunes these parameters to the best possible result. This tool is very good for making quick bandwidth estimations for testing networks and internet speed.

Iperf compatibility with applications is somewhat lacking. *The tool blocks the process until the estimation is complete*, which means that it is not possible to obtain the bandwidth values in real-time in an application. Among the Maximum TCP Throughput tools, Iperf is the only one in which this happens. The probable cause may be the Windows OS binary. The Linux version was tested, however it does not work with the Windows Subsystem Linux. In order to solve the problem, the Iperf tool was adapted by re running every 5 seconds and obtaining the average bandwidth value so far. This will have a particular effect in the bandwidth estimations which is explained in the results section.

4.4.2 Thrulay

Thrulay is a measurement tool developed by Stanislav Shalunov. It performs TCP throughput tests similar to many other tools, but at the same time measures round-trip time. The

version deployed for this thesis was thrulay-hd which is an improvement on the original tool [62]. Throughput metrics are provided by Thrulay-hd with a high degree of consistency and reliability.

4.4.3 Ntttcp

Ntttcp is a Winsock-based tool for Maximum Achievable TCP Throughput similar to Thrulay and Iperf. Ntttcp was developed by Microsoft and it provides a multithreaded, asynchronous performance workload for measuring an achievable data transfer rate on an existing network setup. It is widely used by Microsoft engineers (and, curiously, by the United States of America's Department of Veteran Affairs [63]). The tool has a Windows OS! (OS!) version, however the Linux Version was opted for using via WSL.

4.4.4 Pathrate

Pathrate is an UDP based measurement methodology that can estimate the capacity of Internet paths [64]. This tool uses the packet-pair dispersion and train of packets in order to estimate the capacity. As the size of train of packet increases, the capacity value converges to the ADR (Asymptotic Dispersion Rate). The results that this tool offer are limited to capacity estimation and as such have a limited use for the scenario at hand. Still, Pathrate utilizes two very common techniques for bandwidth estimations which are packet-pair dispersion and train of packets dispersion, so it is worth analysing.

4.4.5 Pathload - Yaz

Pathload and Yaz implement the SLoPS methodology to determine the ABW. Yaz is an improvement over the Pathload tool [65]. Periodic packet streams are generated using UDP-User Datagram Protocol while control channel in-between two terminals is set up by TCP connection. As it is defined in the SLoPS technique, successive probe packets are sent at increasing rate until packet's one way delay increase will cause the queue on the tight link to be temporarily overloaded if the stream rate exceeds the path's available bandwidth. Achieving the max probe rate without generating overload is how SLoPS based techniques estimate bandwidth. Yaz does not use any individual packet spacings when inferring congestion along a path and it instead uses mean spacings. It also uses both stream expansion (i.e., increasing trends in one-way delays) as well as stream compression (i.e., decreasing trends in one-way delays) to infer congestion.

4.4.6 Initial Gap Increasing (IGI)

Within a one-hop network, this tool transmits packet trains with increasing time gaps and evaluates their relationship with competing traffic on the least capacity link. The ABW is estimated as the difference between evaluated bandwidths of competing traffic for each

packet train bitrate and the least capacity link[66][67]. A formula is based on these values to approximate the bandwidth as presented in figure 40.

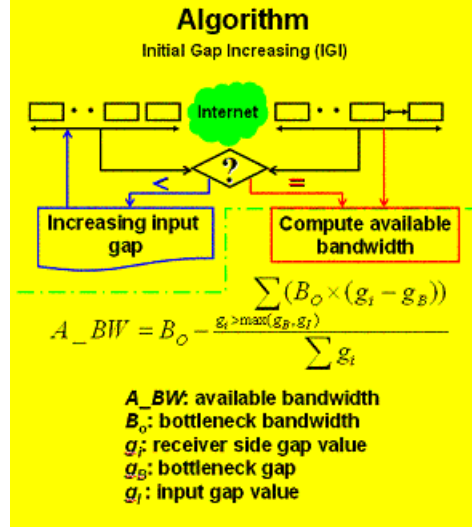


Figure 40: Formula for IGI's Algorithm [68]

4.4.7 Packet Transmission Rate (PTR)

PTR shares the same authors as IGI and works similarly to it. PTR utilizes the same probing formula for bandwidth estimation but IGI focuses on calculating background traffic load, while PTR directly calculates packet transmission rate, to estimate end-to-end available bandwidth [68].

4.4.8 pathChirp - Assolo

pathChirp deploys the SLoPS methodology but features an exponential spaced flight pattern of probes called a chirp. This tool works similarly to other SLoPS tools. The unique feature of this tool is an exponential spaced chirp probing train. For the evaluation of the path's ABW, statistical analysis is performed at the receiver side over the queuing delays, i.e. packet inter-arrival times corresponding to the probe packets transmitted from the sender and finding the probing rate at which queuing starts indicating congestion of network [69]. Assolo is an improvement over pathChirp which implements a novel "Reflected Exponential Chirp", which probes a wide variety of rates, with the center of the probing period being the most accurate [70].

4.4.9 Abing

Abing is another tool based on packet pair dispersion technique. Typically, a train of 10 or 20 closely spaced probes is dispatched to a single target. The assessment of available bandwidth and the evaluation of the observed packet pair delays are based on a technical examination of the obstacles that the frames may encounter in routers or other network devices [5].

4.5 Content Preparation

In order to evaluate the efficiency of the tools presented, it is important to evaluate the effect they will have in the video streaming. Has a such, a preparation of videos for testing was required. SmoothMV already had two multi-view videos available, however, for the context of this thesis, it was not necessary to use multiple views, so new videos were picked. The preparation of the content involved coding the views according to ffmpeg and GPAC's software packages with different segment sizes to optimize their delivery.

4.5.1 Server Video Preparation

The storage of the videos in the servers was prepared as it is illustrated in figure 32. As multiple views weren't used in this work, each bitrate's folder only stored one segmented video identified as view_1. Different videos with different segments sizes were experimented with, namely, 33ms, 100ms 500ms and 1s. For each of the segment's size, it was analysed the time it took to switch the video's quality and the differences were small. However, in the SmoothMV system, smaller segment sizes required a bigger buffer to stream the video without failure.

When working on the video preparation for testing the quality transactions, a significant challenge emerged, which will be presented in this section.

When downloading the video segment in the View Buffering Model, it is necessary to define two buffer sizes, the initial buffer and the playback buffer, with a number of segments. For example, if the initial buffer is defined with a size of 50 segments, the View Buffering will download 50 segments and stored them in the initial buffer; when the video starts playing, the first 50 segments are sequentially played from the initial buffer. The subsequent segments are stored in the playback buffer before being played. This means that if the playback buffer is defined as 50, and the segments stored have a duration of 100ms, if the system is requested to change the quality, the quality will only be effectively switched $50 * 100ms = 5000ms = 5s$ later. Following this logic, it stand to reason that playback buffer should be defined as small as possible, however, reducing too much the playback buffer will result in stutter in the video reproduction. A delicate balance between segment size and playback buffer is required in order to ensure the video plays without interruption and with a reasonable quality switching time.

As it was mentioned previously, the video is played in the GPAC's mp4box video player. This player has the particularity that if it fails to receive a large enough consecutive number of segments to stream, the reproduction stops. This happened often and it proved to be a complicated problem to solve. Nevertheless, a solution was found that mitigated the problem enough in order to make the video streaming with frequent quality switches possible.

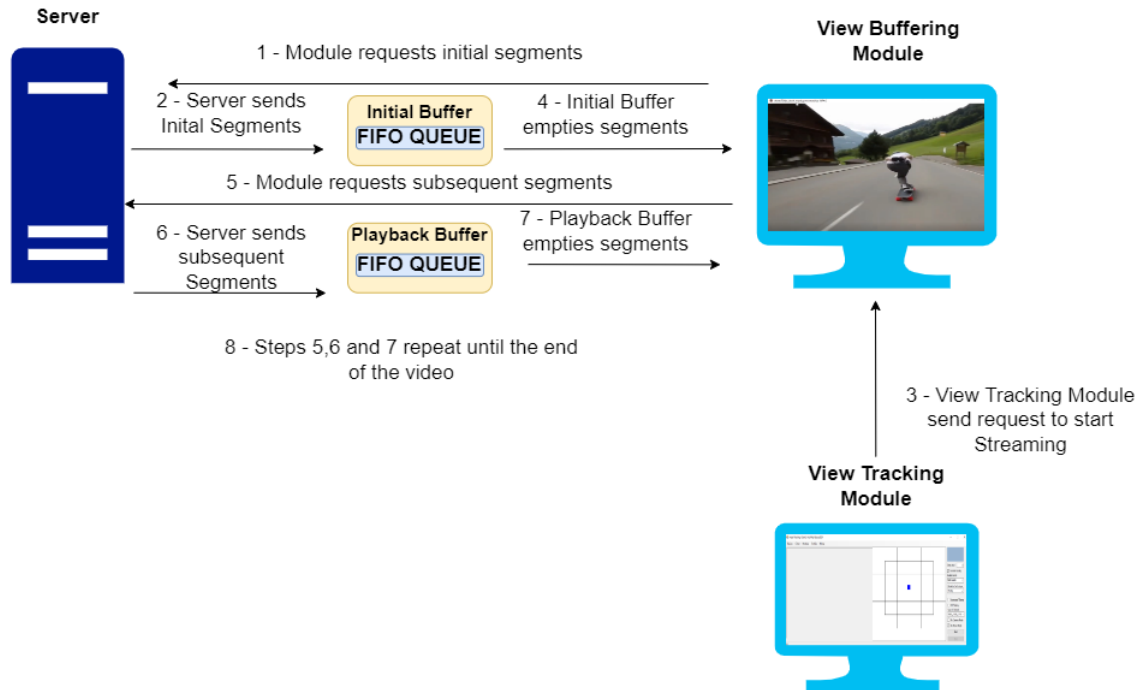


Figure 41: Sequential Server Download

The View Buffering Module downloads the segments sequentially in a single queue. If that queue fails to download a large enough number of segments the video player gives up and freezes. A solution for this was to introduce *multiple downloading processes each running in a different thread*. If the number of threads is defined, for example, as 3, the first thread will download segment N , the second thread will download segment $N + 1$ and the third thread will download segment $N + 2$. The sequence is repeated for the next segments: the first thread would then download segment $N + 3$, the second thread would download segment $N + 4$ and the third would download segment $N + 5$. The View Buffering Module download process was changed in order to accommodate the new mechanism.

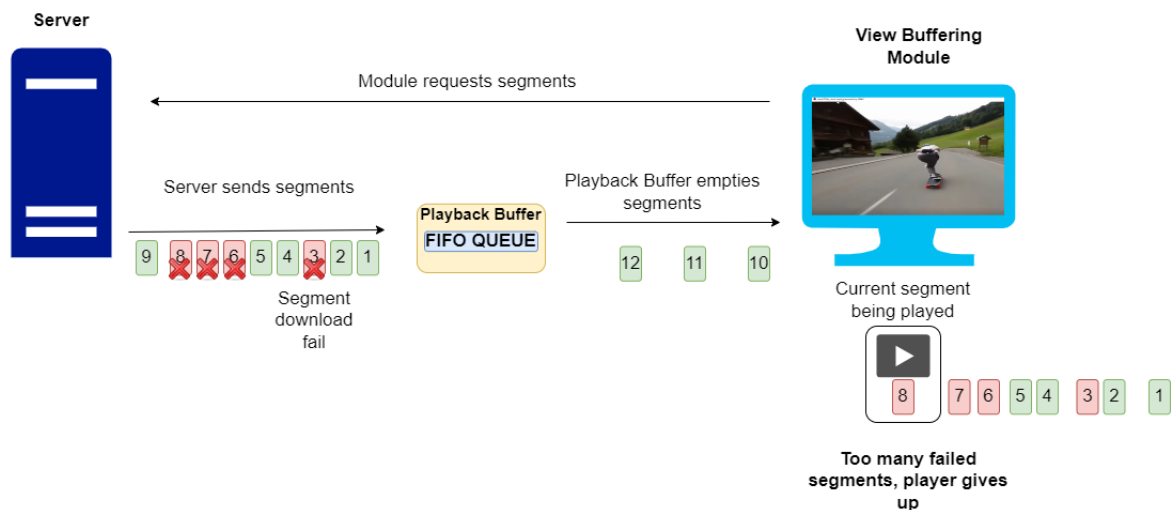


Figure 42: Sequential Server Download

This mechanism reduces the probability that the player is left without segments to fail. Even if a segment fails to download, it is likely that another thread already downloaded another segment, which ensures the continuity of the video. However, this may lead to some frame loss.

It was noted that for smaller segment sizes, 33ms for example, as they required a larger number of segments for the playback buffer, the probability of failure increased.

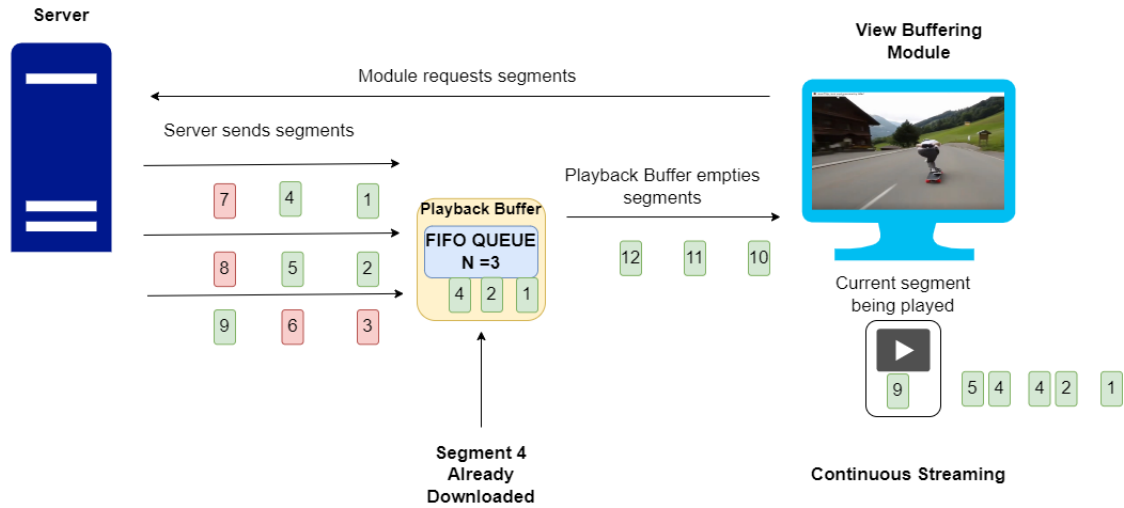


Figure 43: Parallel Server Download

The number of threads is naturally limited by the processing power of the machine. For a high number of parallel threads, the computer starts to have trouble processing them which results in segments failing to download.

4.5.2 Video Content

One concern of this thesis was to examine the effects of quality transactions on the Quality of Experience for various video content with different spatial frequency. The objective was to understand if the perceived quality of the video experience varied significantly for the same bitrate transaction. For example, reducing the bitrate from 5000k to 3500k in a video of a person talking vs a video of a football match.

Three videos were selected with a segment time of 1s and the buffer size of 4 segments, resulting in a 4s time for quality transaction. This combination of segment time and buffer size was the one which produced the lowest switching time at the same time ensuring the continuity of reproduction. The videos selected were encoded in H.265 with 5 different quality level with 24fps:

- Very High Quality - 3500kbps
- High Quality - 3000kbps
- Medium Quality - 2000kbps
- Low Quality - 1000kbps
- Very Low Quality - 500kbps

These values were selected based on experimentation and typical video bitrate for these qualities [71].

The videos chosen were the Canadian's Space Agency "Sleeping in Space" [72], GoPro's 2 "GoPro: Foggy Forest MTB" [73] and Harry Kane Penalty Miss England Vs France World Cup [74].



Figure 44: Frame from Sleeping in Space



Figure 45: Frame from GoPro: Foggy Forest MTB



Figure 46: Frame from Harry Kane's Penalty

The quality adaption was made based on the 5 quality levels. If the bandwidth was above 25 Mbits/s, the player would download very high quality segments, if it was between 5 Mbits/s and 25 Mbits/s (including), High Quality segments were downloaded, if it was between 3 Mbits/s and 5 Mbits/s - Medium Quality, between 1.5 Mbits/s and 3Mbits/s, Low Quality and finally below 1.5Mbits/s the player would download very low quality. These values were based on Netflix's recommendations for video quality depending on the internet bandwidth [76].

Chapter 5

5 Results

5.1 Bandwidth Estimation Tool Objective Performance

In order to evaluate the performance of each tool, several different tests were performed during bandwidth measurements. The values obtained from each tool were analysed and interpreted in a Python's Jupyter Notebook environment. Each tool was evaluated by the 3 metrics presented previously: accuracy, robustness (stability, response time, response to interference) and applicability to the scenario at hand. The effect of the performance of some tools was then evaluated in user's quality of experience by the means of a survey.

Certain effects were observed when deploying certain tools. These effects were then recorded and presented to users in the survey. Users were asked to categorize their video-watching experiences while the videos were subjected to different quality adaption carried out by some tools. The survey results are analysed in the next section.

Here, the strategy was to evaluate each tool and decide whether to keep or reject it depending on how adequate it was for the quality adaption.

5.1.1 Capacity Estimation Tools

The first tools to be classified as unfit for quality adaption were the capacity estimation tools. Capacity estimation can easily justified for network design or for analysing network congestion, however, these tools are a poor choice for video quality adaptation.

As demonstrated by Jacob Strauss and M. Frans Kaashoek in [38], there is a significant difference between the achievable throughput of a link and its capacity, this difference is largely dependent on the speed of the link in question. For very high speed connections, like the SmoothMV's scheme, there is a large discrepancy between the capacity and what is actually possible to transmit in the link. This effect can be visualized in figure 47. Paths with connections with a very large capacity, would ultimately have a very large discrepancy.

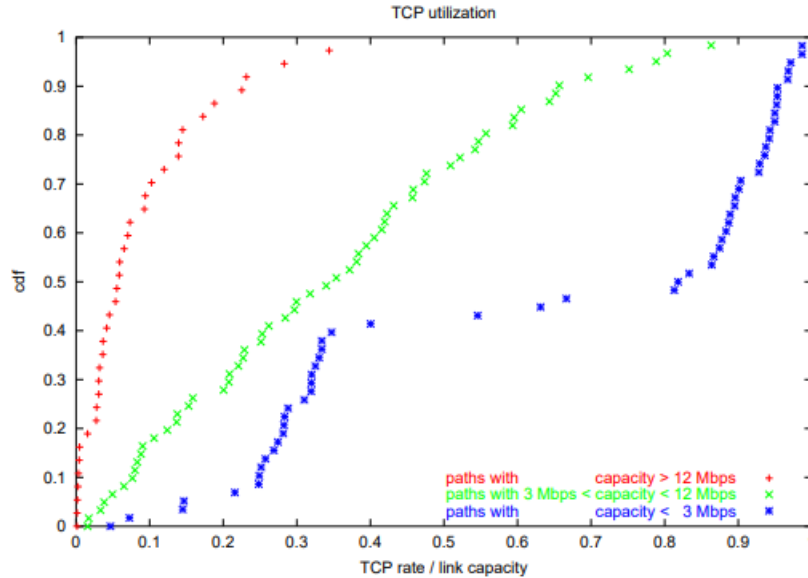


Figure 47: Quocient between Capacity and Bulk Transfer Capacity in different speed links [38]

Comparing Pathrate with all the other tools, it can quickly be observed the large difference in the results. The values obtained in a capacity estimation are not suitable for direct adaption of bitrate based on the bandwidth. Running Pathrate in the link between machine 1 and machine 2, the results were simply "Probably larger than 1000Mbits/s".

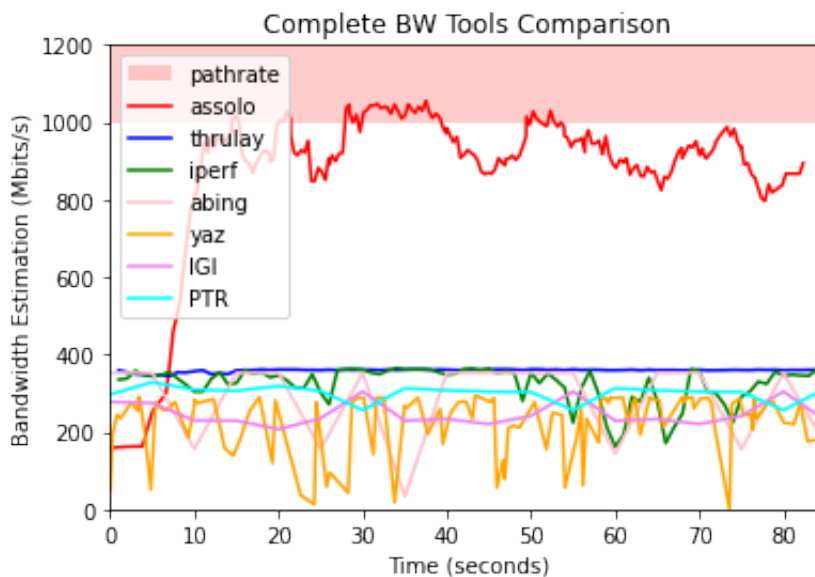


Figure 48: Comparison between all tools without any interference or cross traffic

In Figure 48, the difference of results between Pathrate/Assolo and all other tools is very evident.

The literature suggests Assolo as a technique for estimating the Available Bandwidth, although its results largely overestimate the bandwidth. When it is deployed, the tool produces very low bandwidth results (approximately 70MBytes/s) and issues a warning that the link is extremely fast and that coalescence is being detected. The warning then suggests increasing the size of the "number of packets per Jumbo packet". Increasing the size of the packet cause the tool to largely overestimate the available bandwidth. The theory put forward in this thesis to explain this occurrence is that as the tools send more packet trains, their sensitivity to queueing delays declines until they approach the link's theoretical capacity.

As Pathrate failed to estimate values for a capacity, the accuracy was classified as **low**. Even though it produces a response very quickly, it failed to give any value whatsoever so the robustness was classified as **very low** and the applicability as **very low** as well.

Assolo failed to estimate the available bandwidth but in turn its results approximated the capacity, receiving a **low** accuracy result. The tool takes a long time converging to the capacity values and produces a very large overhead in the network, thus its robustness is **low**. As the tool takes very long to give any results and overly estimates them, it is classified as **very low** in applicability.

5.1.2 Available Bandwidth Estimation Tools

In this category, we can group IGI, PTR, Abing and Yaz. Among these tools, the results were mostly consistent, but there considerable differences in their performance.

As observed in Figure 48, IGI and PTR provided the most stable results. Nevertheless these two tools have the particularity that they only offer a single bandwidth value per measurement. Not only that, but they require some time to obtain results. Both IGI and PTR require at least 5 seconds to get a single result and these still may vary a lot. These tools also use a very small default probing packet size - 500 bytes; the results tend to be low for this packet size. The maximum packet size the tools can handle is about 8000 bytes and even then, the results tend to be slightly under the real bandwidth.

Abing also provides stable results but tends to underestimate the bandwidth. It is also requires at least 5 seconds to produce any result whatsoever.

Yaz returns continuously bandwidth estimation results every second but they vary drastically.

In order to evaluate these tools response to interference, Cross traffic was generated by the iperf tool. The cross traffic's bitrate was increased for each measurement. As the traffic was generated, the bandwidth was measured, at least 5 times, for each of the Available Bandwidth Tools and for the Thrulay tool. As Thrulay gives very stable results, it served as a point of comparison. The aim of this test, besides testing the tools response to interference, was understanding how accurate each tool was in comparison to Thrulay.

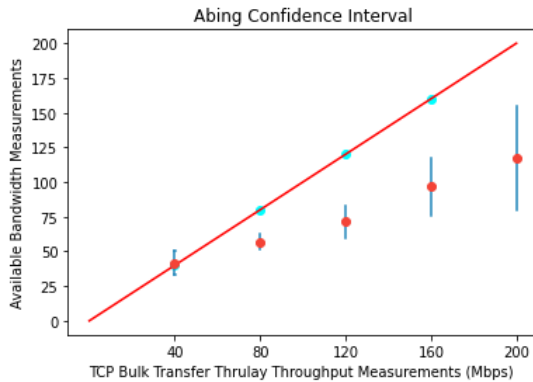


Figure 49: Abing Results for Cross-Traffic

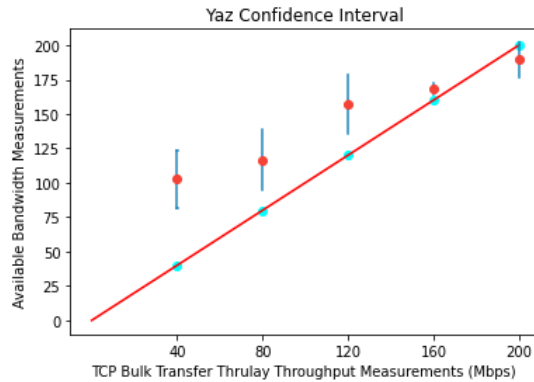


Figure 50: Yaz Results for Cross-Traffic

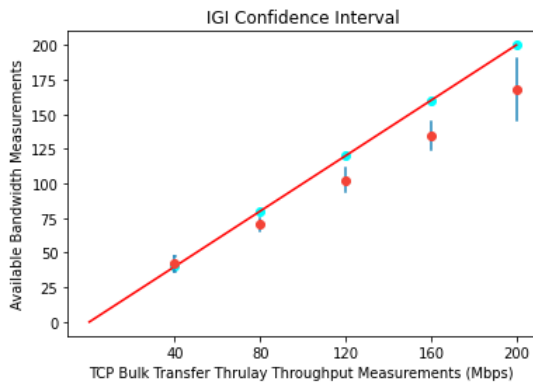


Figure 51: IGI Results for Cross-Traffic

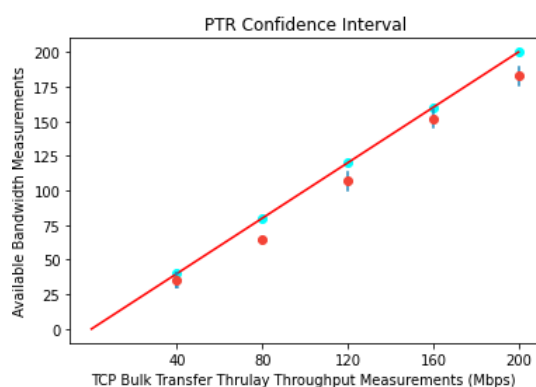


Figure 52: PTR Results for Cross-Traffic

Abing as demonstrated in figure 49, fails to accurately estimate bandwidth for higher values as it tends to largely underestimate them. Not only that, as evidenced by the large interval confidence for higher values, the results tend to vary a lot, going as high as 150Mbps/s and as low as 80Mbps/s, while at the same time, Thrulay measures 200Mbps/s.

Yaz, contrary to what was verified in Figure 48 (when there was no cross-traffic), tends to overestimate the bandwidth values even when a lot of cross-traffic is present. Also, Yaz becomes less stable as cross-traffic increases as it can be observed in the large confidence interval for lower bandwidth measurements.

Both IGI and PTR prove to be more or less stable, however PTR is slightly better. IGI tends to underestimate the bandwidth more for larger bandwidth values and tends to be less stable.

It should be noted that when there was no cross-traffic, none of these 4 tools reached the values that iperf thrulay and ntttcp (all Maximum Achievable TCP tools) measured. The Available Bandwidth tools provided smaller values, around 270Mbps/s - 300Mbps/s, while iperf, thrulay and ntttcp rounded 360Mbps/s.

Client-Based Adaptive Bitrate Streaming algorithms usually calculate the bandwidth as the size of the fetched segment(s) divided by the transfer time - throughput-based. This means if there is a failure in packet transmission, the measured bandwidth is reduced. This proves the importance of the bandwidth measurements reflecting the data reaching the streaming platform. A very important aspect of the Available Bandwidth Estimation (ABW) algorithms is that *they are not sensitive to network limitations*. This means that if, for some reason, the server, the host that's sending the packets, is temporary limited in upload, the tools won't reflect that in their estimations. ABW methodologies like Variable Packet Size, Packet Pair /Train Dispersion, Self-Loading Periodic Stream and Trains of Packet Pair aren't based on upload throughput as opposed to TCP Throughput Tools like Iperf, Nttcp and Thrulay.

This phenomenon was observed by imposing an upload limitation of 1 MByte/s to the Local Network via NetLimiter [75]. Assuming that both hosts are in the same LAN, the maximum bitrate possible which the server could send to the client would be 1MByte/s after the limitation. This also means the available bandwidth for the video streaming would be at best 8Mbits/s - 1Mbyte = 1 Mbit.

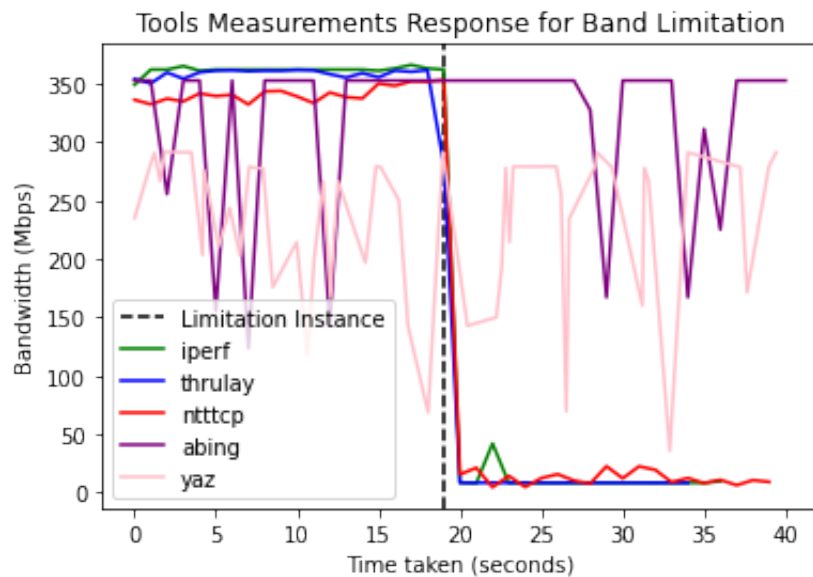


Figure 53: Comparison between tools response to upload limitation

Figure 53 shows the results produced by ABW tools and Bulk Transfer Capacity tools after limiting the upload to 1MByte/s. The estimations by the ABW tools don't reflect the limitation at all. This can be explained by the fact that the ABW Estimation methodologies are based on mathematical approximations of the bandwidth based on variables independent to throughput like packet dispersion and sending rate. This phenomenon reduces by a lot the ABW tool's applicability to the scenario at hand.

5.1.3 TCP Throughput and Bulk Transfer Capacity

The tools that proved to be the most applicable to the SmoothMV were the TCP Throughput based tools. They make very fast, stable and continuous bandwidth estimations and they are capable of reflecting upload limitations. However, it should be noted that even these tools aren't at all perfect. They are the most intrusive of all the tools and even though they are to reflect upload limitations, their calculations are not absolutely correct. If the throughput is limited to a maximum of 1Mbyte/s, than the bandwidth between server and host should at most 8Mbit/s, but they overestimate slightly as it demonstrated in figure 54.

```
C:\Users\pelay\Desktop\tools\iperf-3.1.3-win64>iperf3 -c localhost -t 192.168.1.101
Connecting to host localhost, port 5201
[ 4] local ::1 port 63645 connected to ::1 port 5201
[ ID] Interval           Transfer     Bandwidth
[ 4] 0.00-1.00      sec   39.9 MBytes  333 Mbits/sec
[ 4] 1.00-2.01      sec   18.1 MBytes  151 Mbits/sec
[ 4] 2.01-3.02      sec   18.1 MBytes  151 Mbits/sec
[ 4] 3.02-4.00      sec   19.9 MBytes  169 Mbits/sec
[ 4] 4.00-5.01      sec   19.5 MBytes  163 Mbits/sec
[ 4] 5.01-6.00      sec   19.1 MBytes  162 Mbits/sec
[ 4] 6.00-7.01      sec   19.6 MBytes  164 Mbits/sec
[ 4] 7.01-8.01      sec   19.6 MBytes  164 Mbits/sec
[ 4] 8.01-9.00      sec   19.4 MBytes  164 Mbits/sec
[ 4] 9.00-10.00     sec   15.6 MBytes  131 Mbits/sec
[ 4] 10.00-11.01    sec    1.00 MBytes   8.34 Mbits/sec
[ 4] 11.01-12.01    sec    1.00 MBytes   8.36 Mbits/sec
[ 4] 12.01-13.01    sec    1.00 MBytes   8.37 Mbits/sec
[ 4] 13.01-14.00    sec    1.00 MBytes   8.50 Mbits/sec
[ 4] 14.00-15.01    sec    1.00 MBytes   8.35 Mbits/sec
[ 4] 15.01-16.01    sec    1.00 MBytes   8.34 Mbits/sec
```

Figure 54: Iperf's response to upload limitation

Since these tools were the best ones so far, in order to compare them, they were tested by analysing the effect they had on the video adaption.

As it was previously mentioned, 5 video quality level were used, ranging from Very High Quality to Very Low Quality. Depending on the bandwidth value, the video segments would be download with the quality corresponding to its interval. The ranges for each quality were chosen based on Netflix's Video streaming guidelines. The performance of the TCP Throughput tools was evaluated by how quickly they responded to bandwidth variations, how stable the values estimated were and how well the estimations fitted in each quality interval ; ie, if a tool provides results which vary a lot between quality intervals, the video quality will switch excessively and impact the Quality of Experience.



Figure 55: General Required Bandwidth for Streaming Guidelines [77]

Iperf produced results which were very stable. However, as it was mentioned before, this tool had the particularity that *it only returns all of the results after the estimation process is complete*. Iperf doesn't allow the user to parse the results as they are estimated each second like other TCP-Throughput tools. In order to solve this problem, the iperf process was scheduled to be restarted every 5 seconds and the average value so far was saved as the BW estimated. This means that the system only had access to the values every 5 seconds. The 5 seconds were chosen because it was the average time required for most tools to produce results, and more than 5 seconds proved to be too much time to get estimations. Iperf has also the particularity that it requires some time to stabilize the bandwidth estimations, approximately 5s to 8s to achieve total stability. The estimations start by being slightly higher than the actual bandwidth and then start to steadily stabilize. The time to stabilize varies between each estimation process, which means that *the obtained Iperf may have results slightly above the actual available bandwidth*.

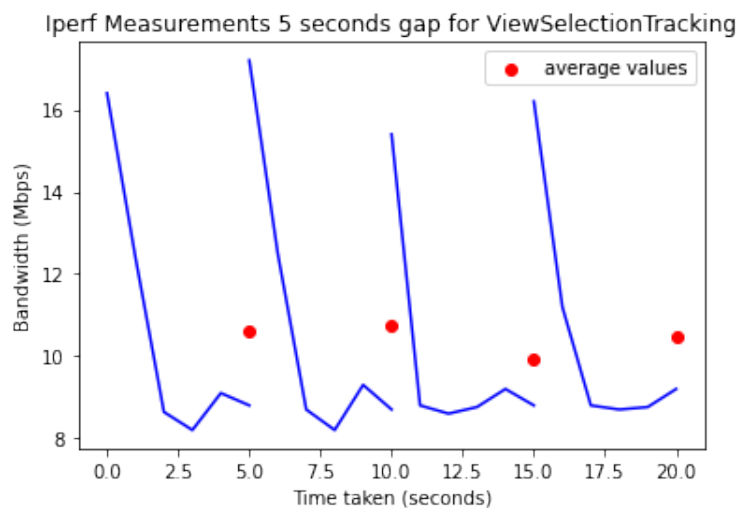


Figure 56: Iperf Measurements every 5 seconds

In figure 56 the iperf overestimation is evident. in this case the actual bandwidth measured should be 8Mbits/s. Iperf starts stabilizing slightly after 4s, but then process restarts at the 5th second. Iperf returns the average value after each estimation process, but as the process doesn't have time to converge, the estimated average will be slightly over the 8Mbits/s.

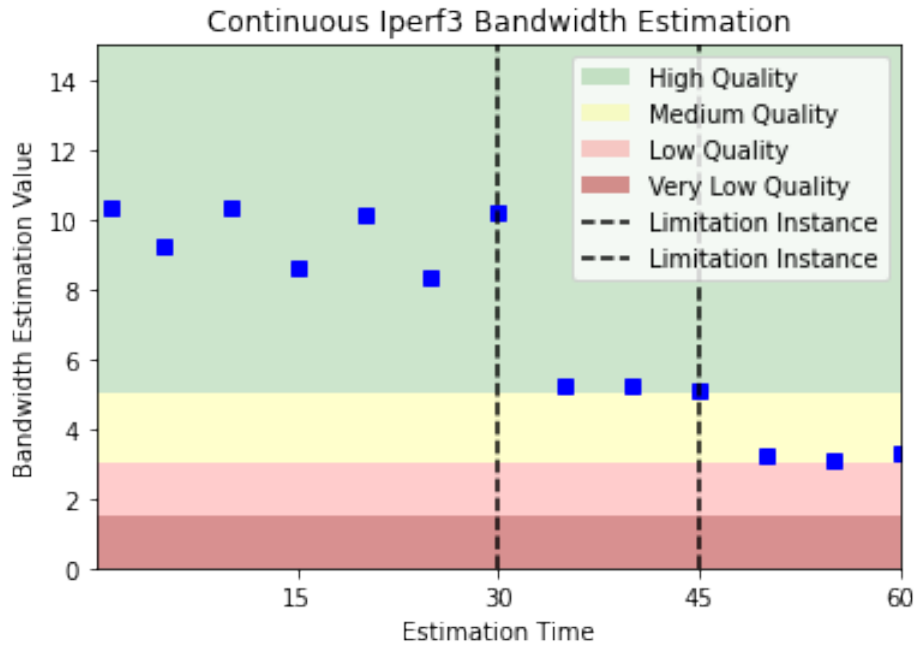


Figure 57: Iperf's response to upload limitation

Figure 57 shows the effect that this slight over estimation has on the quality downloaded. At $t=30$ s, a limitation of 500Kbytes/s to the upload was imposed. This should result in an estimated bandwidth of 4Mbits/s, 500Kbytes = 4Mbits. However, Iperf slightly overestimates the bandwidth as approximately 5Mbits/s, which results in the segments being downloaded with high Quality (as the obtained measurements fit in the high quality interval) instead of medium quality. Similarly, at $t=45$ s, a limitation of 250KBytes/s was imposed which should result in a measured bandwidth of 2 Mbytes/s, nevertheless, once again, the Iperf results are slightly above the actual bandwidth, resulting in segments being downloaded with Low quality instead of Very Low Quality. The overestimation of the values results in the video downloading segments with qualities superior to what the network is able to provide, leading to frame loss and video freezing. The video quality was altered based on the Iperf measurements which resulted in the frame loss as presented in Figure 58.

It should be noted that some frame loss occurs not because of the overestimation of the bandwidth but because of typical errors in download. Still, in figure 58 it is clear that after the quality switches to a level above what the network can provide, there is more frame loss.

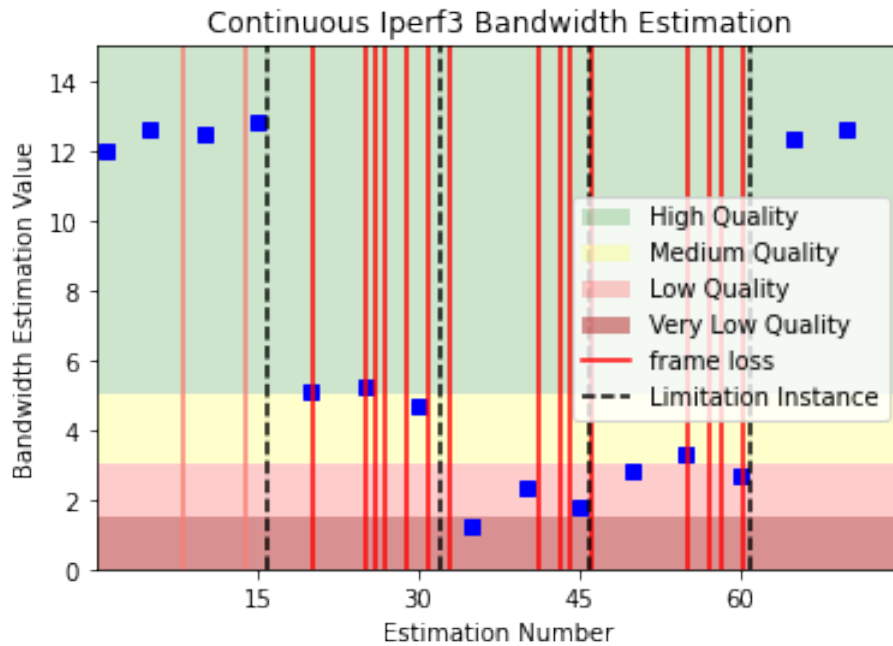


Figure 58: Frame Loss resulting from Iperf Estimations

As for Ntttcp, this tool proved to be more stable for *High Bandwidth Measurements*, specially for bandwidth over 100Mbits/s. In this range Ntttcp tend to be produce somewhat steady values, with a confidence interval of about 6 Mbits/s. However, the quality transactions are issued when the bandwidth is around much lower values. Not only that but a slight overestimation of 8 Mbits/s instead of 5 Mbits/s, for example, may result in the player downloading segments with quality above what the network can handle.

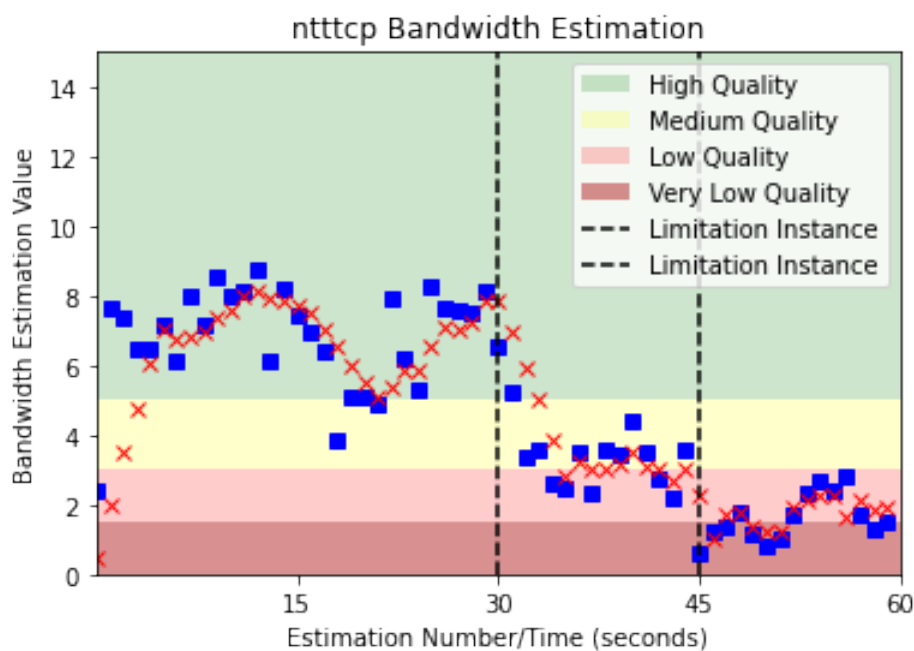


Figure 59: Ntttcp response to upload limitations

A similar test was conducted for Ntttcp's response to upload limitations as it can be observed on figure 59. Ntttcp provides new estimation every second and it allows the user to parse the values as they are estimated unlike Iperf.

Ntttcp is very unstable for low bandwidth, and as a result, the estimations constantly switch between quality intervals. In order to attempt to mitigate this effect, a moving average function of 5 elements was added to stabilize the values, but even so the results are too variable to be viable for video quality adaptation.

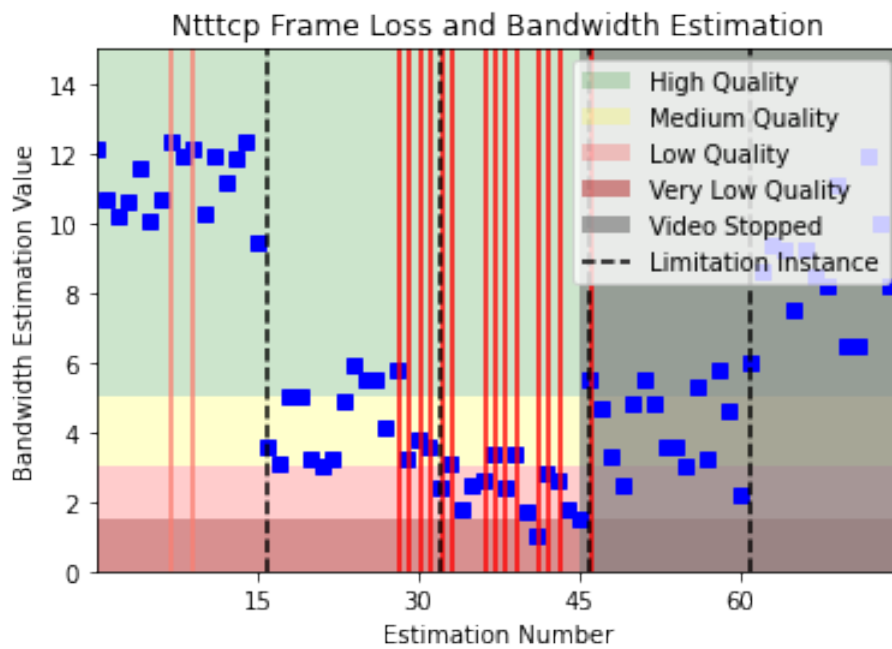


Figure 60: Frame Loss resulting from Ntttcp Estimations

The video froze so often that the player gave up on reproducing it after some time.

Finally, Thrulay proved to be a much more suitable tool than the other two. The estimations obtained oscillated very slightly and landed precisely on the expected bandwidth values. Not only that but Thrulay, like Ntttcp, provided the measurements every second, which is very important to avoid delayed quality transactions.

Also, the quality transactions based on the Thrulay measurements resulted in much lesser frame loss and barely any video freezing whatsoever.

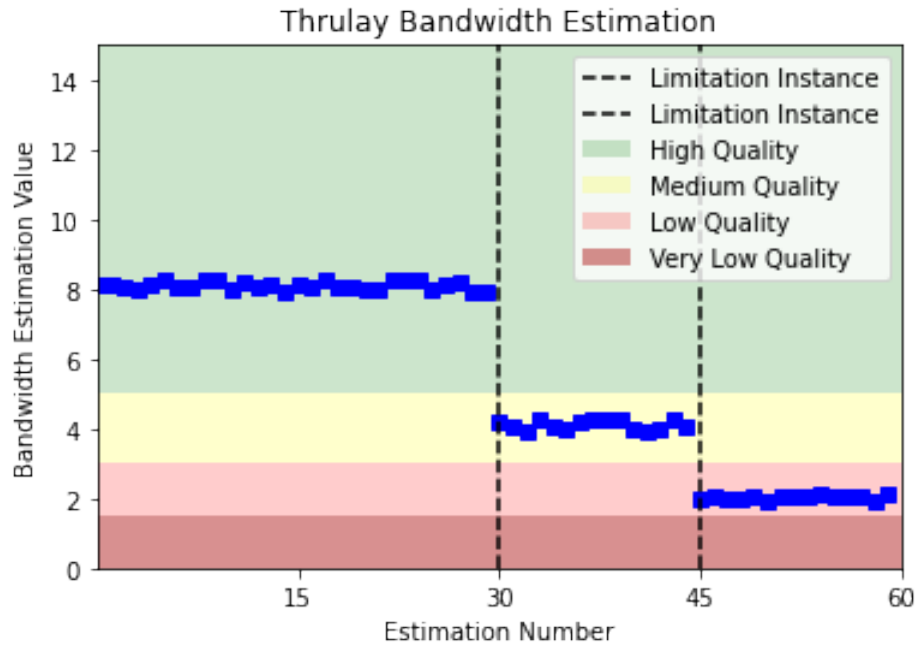


Figure 61: Thrulay response to upload limitations

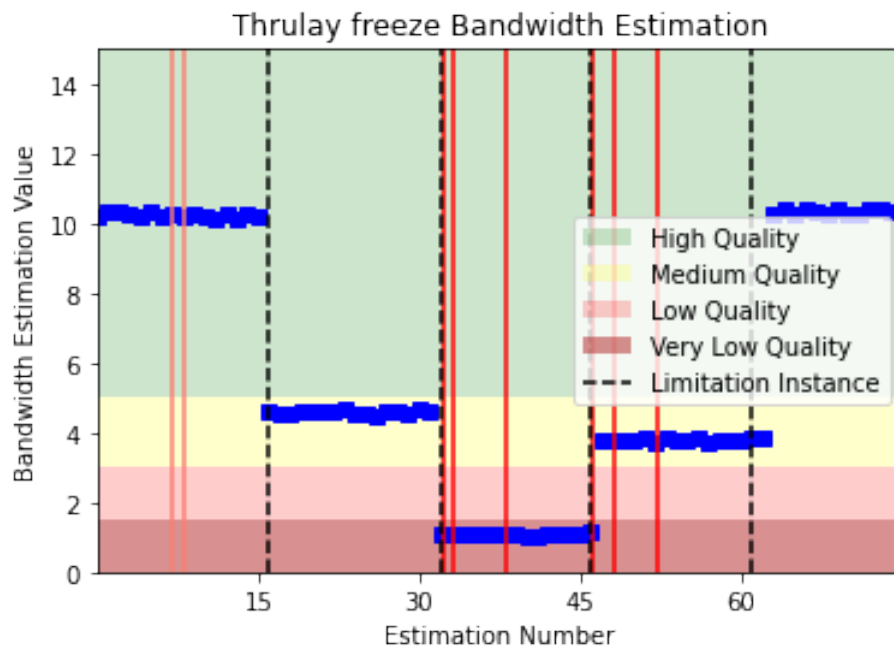


Figure 62: Frame Loss resulting from Thrulay Estimations

5.1.4 Tools Performance Classification

After analysing the performance of each tool, they were evaluated as presented in Table 1.

Tool	Measurement Metric	Accuracy	Robustness	Applicability
pathRate	End-to-End Capacity	Low	Very Low	Very Low
Assolo	End-to-End Capacity	Low	Low	Very Low
Yaz	End-to-End Available-BW	Medium-High	Medium	Medium
Abing	End-to-End Available-BW	Medium-Low	Low	Low
IGI	End-to-End Available-BW	Medium	Medium	Medium
PTR	End-to-End Available-BW	Medium-High	Medium-High	Medium
Iperf	Achievable TCP throughput	Medium-High	High	Medium-High
Ntttcp	Achievable TCP throughput	Medium-High	Medium	Medium-Low
Thrulay-hd	Achievable TCP throughput	Very High	Very High	High

Table 2: Measurement result of Exp.1

Final Note - The evaluation of each of these tools was naturally biased to the SmoothMV scenario. Just because one of the tools was scored as low Robustness or low Accuracy in these experiments doesn't mean that it won't score higher in a different scheme. For example, pathRate failed to measure the link's capacity due its very high speed, however if the connection was slightly slower it would probably measure the capacity correctly.

It should also be noted that it could be argued that comparing some of these tools directly is unfair. Even though that all these tools aims to approximate the abstract concept of "bandwidth", the metric they measure is effectively different.

5.2 User's Quality of Experience

The effects that the bandwidth estimation tools produced in the video streaming were also analysed in a more subjective User's Quality of Experience point of view. The aim of this study was also understanding how user perceived the video reproduction experience after being subjected to different interferences and quality degradations in order to find a correlational link between these and viewer's attitude, similarly to the study conducted by Shunmuga Krishnan and Ramesh K. Sitaraman.

5.2.1 Structure of the Survey

A good amount of users would be needed to obtain robust results. A Google Form was prepared and sent via email was to friends, family and colleagues explaining the premise of the study and inviting them to participate and help gather data. The survey managed to gather 42 answers after 2 weeks.

The videos presented to the user were obtained by recording the screen while the videos were being streamed in the View Buffering Module. Later, in order to make them easily accessible, these were posted to YouTube.

The Form presented users with the videos posted on Youtube and subsequently asked some questions in order to classify their experience.



Figure 63: Video presented to users via Youtube's Link

5.2.2 Video Streaming for different Available Bandwidth

The first section of the form presented users with the Canadian ´s Space Agency ´Sleeping in Space´ streamed in the View Buffering Module with High Quality, Medium Quality, Low Quality and Very Low Quality. Each of these scenarios represented the video being streamed for different Available Bandwidths - High Quality having the most ABW and Very Low Quality having the least ABW. All of the videos started by played with High Quality - 3000kbps and the qualities were switched according to Thrulay estimations. For each case the quality oscillated in the following manner:

- **Highest Quality** - *Very High Quality - 3500kbps* and *High Quality - 3000kbps*;
- **Medium Quality** - *Medium Quality - 2000kbps*, occasionally *High Quality - 3000kbps* and occasionally *Low Quality - 1000kbps*;
- **Low Quality** - *Low Quality - 1000kbps*, occasionally *Medium Quality - 2000kbps* and occasionally *Very Low Quality - 500kbps*;
- **Veru Low Quality** - *Very Low Quality - 500kbps*, occasionally *Medium Quality -2000kbps* and occasionally *Very Low Quality - 500kbps*;

The bandwidth was reduced by imposing limitations in the upload via NetLimiter. After each video, the user was requested:

- To rate the quality of the image as ***Very Bad, Bad, Average, Good*** and ***Very Good***;
- To rate how much did the quality adaptation affect the experience from a scale of 0 to 10, 0 ***Didn´t Affect the Experience at all*** and 10 being ***Completely Ruined the Experience***;
- To rate the notoriety of interruptions in the video (stutter, frame loss and starting delays) as ***Not Noticeable, Almost Not Noticeable, Noticeable, Very Noticeable*** and ***Extremely Noticeable***;
- To rate how much did the interruptions affected the experience from a scale of 0 to 10, 0 ***Didn´t Affect the Experience at all*** and 10 being ***Completely Ruined the Experience***;

The distribution of the results for the first question ´How would you classify the video quality?´ can be visualized in the following circle graphs:



Figure 64: Correspondence between Color and Answer for the question ´How would you classify the video quality?´

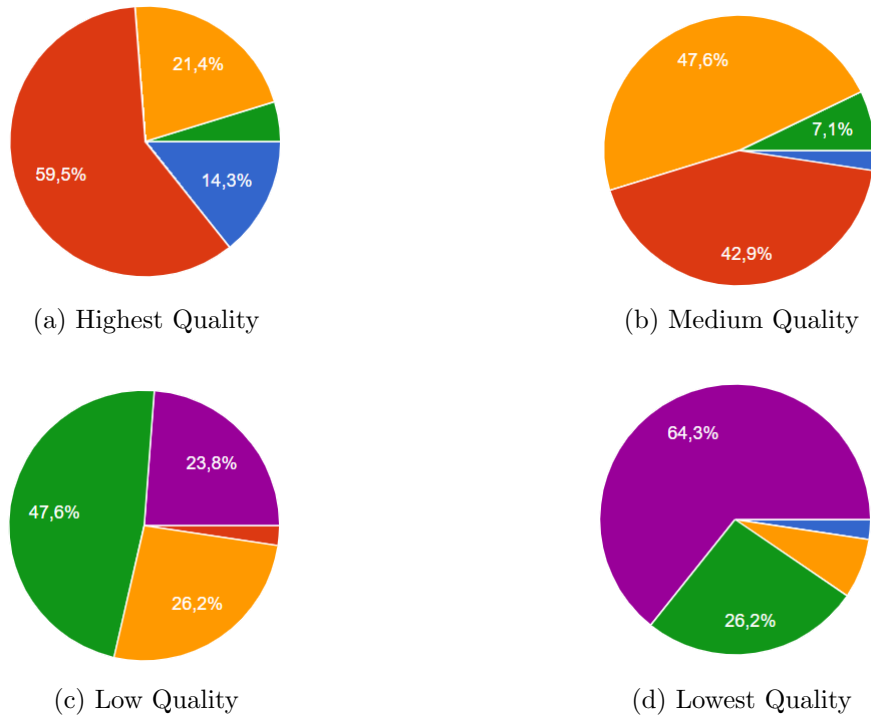


Figure 65: Viewers answers to the Question: "How would you classify the video quality?"

The answers obtained proved what was already expected, which is *Viewers are very sensitive to video bitrate*. This is clearly observed, for example, in the answers obtained for the highest quality; even for the best quality encoded, about 25% of the viewers classified the image quality as average or low. Not only that but it seems that the viewers really disliked the low quality and the very low quality, the vast majority didn't even classify these as average.

As for the second question "How much did the quality transactions affect your video experience?":

- *The Highest Quality obtained an average of: 3.3;*
- *The Medium Quality obtained an average of: 4.3;*
- *The Low Quality obtained an average of: 6.6;*
- *The Lowest Quality obtained an average of: 8.0;*

Viewers clearly didn't enjoy when the video quality degraded to the lower qualities. Most of them replied that it completely ruins the experience .

The distribution of the results for the first question "How noticeable were the interruptions in the video?" can be visualized in the following circle graphs:

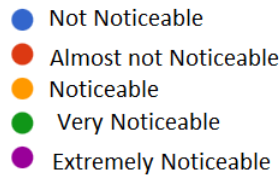


Figure 66: Correspondence between Color and Answer for the question "How noticeable were the interruptions in the video?"

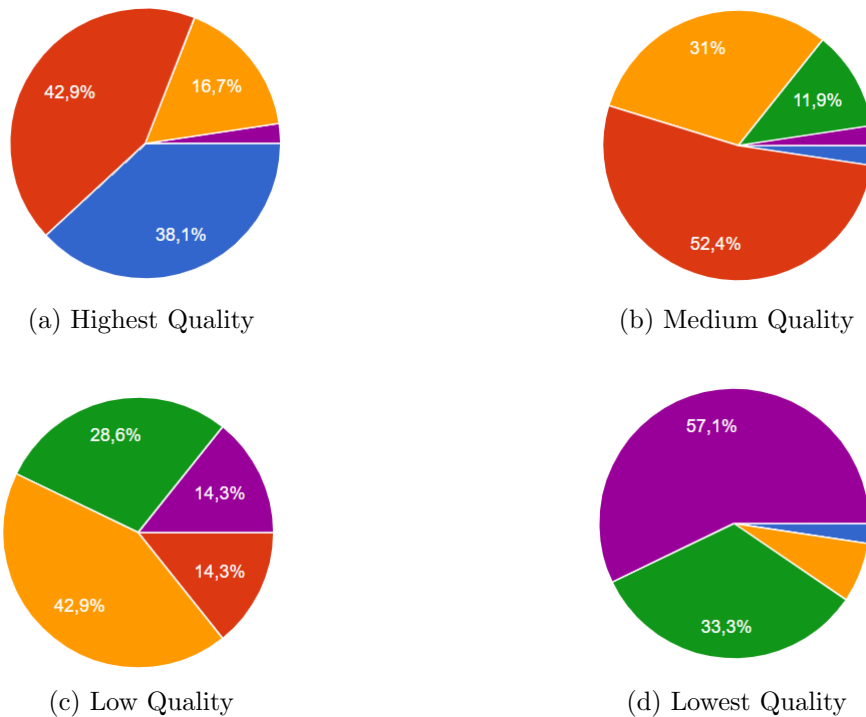


Figure 67: Viewers answer to the Question "How noticeable were the interruptions in the video?"

Here it is clearly visible that when there's quality degradations due to sudden bandwidth limitations, the buffering effect is clearly more present for smaller bandwidths. This is likely a result of the fact that when the amount of available bandwidth suddenly drops to a very low number, there's a small time interval when the quality hasn't been adapted yet and network can't send the higher quality frames.

As for the last question "*How much did the interruptions affect your video experience?*":

- The Highest Quality obtained an average of: **3.42**;
- The Medium Quality obtained an average of: **3.26**;
- The Low Quality obtained an average of: **6.7**;
- The Lowest Quality obtained an average of: **8.5**;

5.2.3 Abrupt vs Smooth Quality Reduction

How the "speed" of the quality decline influenced user experience was one of the aspects that was planned to be evaluated in this study. As such, users were asked to rate two different quality reductions from High Quality to Very Low Quality. The first one was a reduction from High Quality straight to Very Low Quality and the second one was a gradual reduction from High Quality to Very Low Quality passing through Medium Quality and Low Quality.

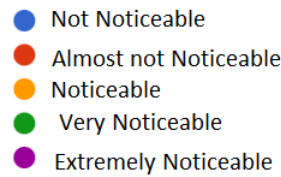


Figure 68: Correspondence between Color and Answer for the question "How noticeable was the Quality Reduction in the video?"

Users clearly preferred the Smooth Quality Reduction and found the Abrupt quality reduction much more noticeable.

When asked how much did the interruptions affected the experience from a scale of 0 to 10, 0 *Didn't Affect the Experience at all* and 10 being *Completely Ruined the Experience* for each quality reduction, the abrupt one received an average score of **7.52** and the smooth of **5.5**. This seems to indicate that both quality reductions negatively impacted the experience, however the abrupt one had a worse impact.

Users were then asked to chose which of the quality reductions had preferred and 95.2% indicated that they preferred the smooth reduction.

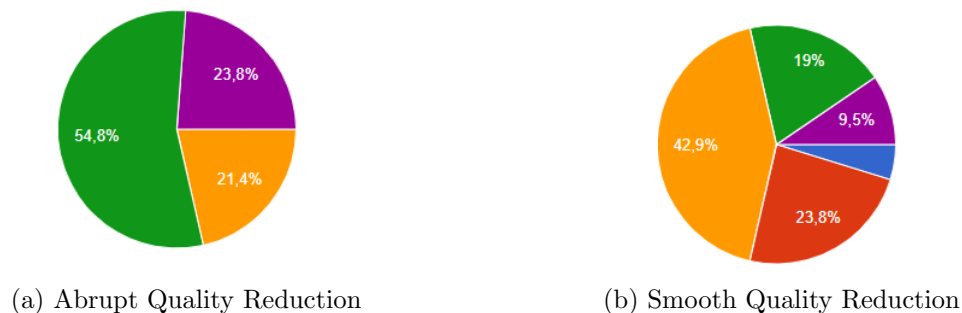


Figure 69: Viewers answer to the Question "How noticeable were the interruptions in the video?"

5.2.4 Quality Reduction for Different Contents

In the third part of the survey, users were shown two different video GoPro's "GoPro: Foggy Forest MTB" and Harry Kane Penalty Miss England Vs France World Cup. For each of these, the video was shown in two different scenarios, *Very High Quality - 3500kbps* with no reduction and *Very High Quality - 3500kbps* reduced to *Very Low Quality - 500kbps*. Both videos were encoded with the exact same bitrate, and both were subject to the quality reduction.



Figure 70: Correspondence between Color and Answer for the question "How would you classify the video quality?" for videos with different content.

The users were asked for a second question "How much did the quality affect your video experience?" and asked to rate from a scale from 0 to 10, 0 *Didn't Affect the Experience at all* and 10 being *Completely Ruined the Experience*; The GoPro Foggy Bicycle DownHill video received an average score of **4.9** and the Harry Kane's penalty had a score of **5.4**. The results, as seen in figure 71, seem to indicate that between the two videos, the results were more or less the same; the perceived quality was average. However, comparing the perceived quality from these two videos with the Canadian Space Agency's "Sleeping in Space" (which was analysed before), it is obvious that that video is perceived as having much better quality compared to the other videos (most user said that the video quality was good and the average score for the quality was **3.3**). The reason for this affect can be attributed to the non liner relationship between bitrate and the Structural SIMilarity plus (SSIMplus)'s, perceptual quality for different types of contents as demonstrated Krishnan and Sitaram, figure 22.

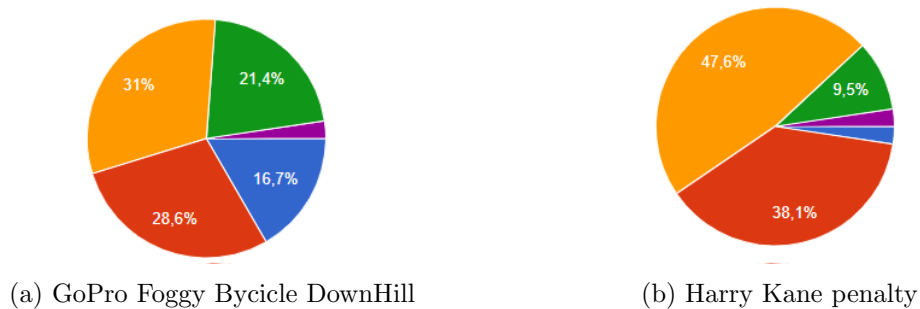


Figure 71: Viewers answers to the Question: "How would you classify the video quality?" for videos with different content.

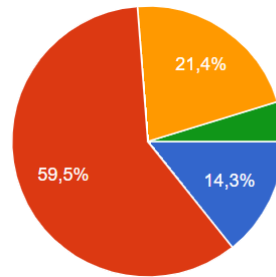
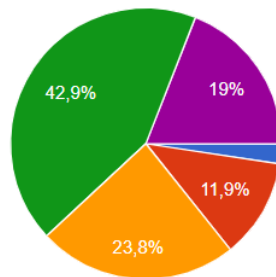
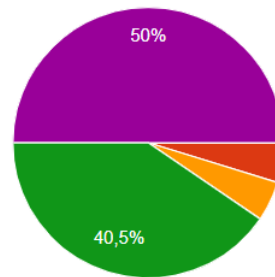


Figure 72: Viewers answers to the Question: "How would you classify the video quality?" for Canadian Space Agency's "Sleeping in Space" with highest quality

The survey then presented users with the Gopro's Bicycle Downhill and Harry's Kane penalty with very low quality. Before the videos were shown, users were asked to review the "Sleeping in Space" video with low quality.



(a) GoPro Foggy Bicycle DownHill



(b) Harry Kane penalty

Figure 73: Viewers answers to the Question: "How would you classify the video quality?" for videos with different content with lowest quality.

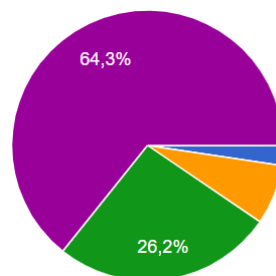


Figure 74: Viewers answers to the Question: "How would you classify the video quality?" for Canadian Space Agency's "Sleeping in Space" with lowest quality

The users were once again asked "How much did the quality affect your video experience?" and asked to rate from a scale from 0 to 10, 0 *Didn't Affect the Experience at all* and 10 being *Completely Ruined the Experience*; and the GoPro Foggy Bicycle DownHill video received an average score of **6.1** and the **6.42**.

Users classified very badly both videos with the very low quality but the football penalty was classified as slightly worse. However, users classified the Sleeping in Space video as specially bad quality even though the three videos were reduced to the exact bitrate. This can be probably be explained by the fact that in that the Sleeping in Space

video there is a pixelization effect in a person's face, which probably contributes more to the perception of "bad quality".

This results seem to indicate that quality deterioration has a more negative impact on videos where detail is more important. If the quality reduction makes it difficult for the user to notice where the football is in a penalty score, this will clearly impact more the user's QoE.

Interestingly, users seem to be more sensitive to the quality for videos where the center of the scene is a person. Users classified the "Sleeping in Space" video with **Very High Quality -3500k** better than the other two videos with **Very High Quality - 3500k**, but at the same time classified the "Sleeping in Space" video with **Very Low Quality -500k** worse than the other two videos with **Very Low Quality - 500k**.

5.2.5 Stutter and Frame Loss caused by Iperf and Ntttcp

Finally, as a way to assert how Iperf and Ntttcp would affect the user's experience, these were put to the test by presenting the "Sleeping in Space" video to users while being adapted by each of these techniques. Iperf produced the effect of downloading segments of higher quality than what the network could provide and Ntttcp produced intense variations in the quality, both of these effects were already observed in the past section.



Figure 75: Correspondence between Color and Answer for the question "How would you classify the video experience?"

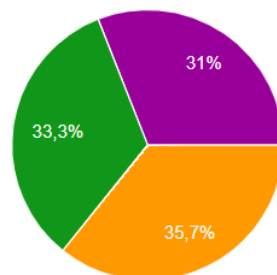


Figure 76: Viewers answers to the Question: "How would you classify the video experience?" for Canadian Space Agency's "Sleeping in Space" adapted by Ntttcp

The users were asked as a final question "How much did the interruptions affect your video experience?" and asked to rate from a scale from 0 to 10, 0 **Didn't Affect the Experience at all** and 10 being **Completely Ruined the Experience**; the Ntttcp adapted video received an average score of **7.4** and the Iperf adapted video **8.2**.

Even though Ntttcp adaption was more "random" than Iperf, this tool's adaption managed to download more low quality segments than Iperf, which consistently overesti-

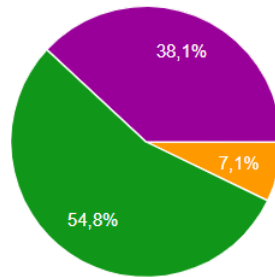


Figure 77: Viewers answers to the Question: "How would you classify the video experience?" for Canadian Space Agency's "Sleeping in Space" adapted by Iperf

mated the bandwidth resulting in constant stutter and frame loss. Ntttcp also had a lot of interruptions, however it managed to reduce to the quality of the video enough time to make the video behave slightly better.

Iperf was classified in the previous section as having a *Medium-High* applicability in the scenario at hand, however after being tested adapting a video and having viewers commenting on its performance, it proved to be less applicable than originally thought.

Chapter 6

6 Conclusion

In this thesis, a Network-based Adaptive Bitrate Streaming system was designed and implemented in the SmoothMV system. Several obstacles arose throughout the project, but they were all handled and resolved as well as possible.

A study was conducted on the most popular active bandwidth tools and how they behave in a video streaming scenario. They were evaluated by their accuracy, robustness and applicability to SmoothMV. Among these, the Maximum Achievable Throughput tools proved to be the most adequate. A comparative study among some of the most used Bulk Transfer Capacity tools was conducted, analysing in particular Iperf, Ntttcp and Thrulay. Iperf proved to be very precise and stable for quick bandwidth estimation between two hosts, however, its inability to produce accessible results in real-time makes it very a complicated tool to implement successfully for video streaming. Ntttcp produced results that varied too much to make it a reliable option. Thrulay's very stable and precise results, as well as the tool's ability to produce results in real-time, makes it the overall best tool to be utilized for adapting the SmoothMV's video quality.

A survey was conducted in order to understand how viewers perceived their video watching experience for different quality reductions and adaptations. The results indicated that the bitrate reduction affected drastically the Quality of Experience, not solely because of the quality reductions but mostly due to the frame drops and video stuttering that comes along with them. Additionally, users appear to favor steady bitrate reductions over sudden quality losses. The survey also served as a way to study how different video contents affected the user's Quality of Experience. Users demonstrated a greater sensitivity to bitrate adaptations of videos with fine details, particularly videos with people in the foreground. The Bulk Transfer Capacity tools were also put to test in the survey, and Iperf and Ntttcp both failed to effectively adjust the video's bitrate without noticeably degrading the quality of the experience for the viewer.

6.1 Future Work

Some aspects of the designed system proved to be sub par. Since the bandwidth estimations need to be processed through an additional layer, starting the estimations on both sides necessitates extra effort. Also, the system doesn't take into account other aspects that may influence the bandwidth estimations. Observing traffic already present in the network and then estimating the bandwidth of the network based on local consumption of bandwidth may be a more adequate approach. Not only that but a mathematical model-based approach that predicts the service in packet network may help the system evaluate the network's performance in a more precise way. This approach requires a significant

amount of effort to explore, but the trade-off may be a significantly more robust Adaptive Bitrate Streaming system.

7 References

- [1] J. Stoll, "Video Streaming Worldwide - Statistics and Facts" Available in: <https://www.statista.com/topics/7527/video-streaming-worldwide>. Last Access: December 29, 2022.
- [2] Grand View Research, "Video Streaming Market Size, Share and Trends Analysis Report By Streaming Type, By Solution, By Platform, By Service, By Revenue Model, By Deployment Type, By User, By Region, And Segment Forecasts, 2022 - 2030" Available: <https://www.grandviewresearch.com/industry-analysis/video-streaming-market> Last Access: December 29, 2022.
- [3] Mansell Robin and Foresta Don, "Social value of high bandwidth networks: creative performance and education " Phil. Trans. R. Soc. A. 374:2015012 420150124.
- [4] Mukta, M. and Gupta, N. (2017). Bandwidth Estimation Tools and Techniques: A Review. International Journal of Research, 4(13), 1250-1265
- [5] Goldoni, Emanuele and Marco Schivi. "End-to-End Available Bandwidth Estimation Tools, An Experimental Comparison." Traffic Monitoring and Analysis (2010).
- [6] Salcedo, D., Guerrero, C. and Martínez, R. (2022). "Available Bandwidth Estimation Tools Metrics, Approaches and Performance" International Journal of Communication Networks and Information Security (IJCNIS), 10(3). <https://doi.org/10.17762/ijcnis.v10i3.3516> (Original work published December 19, 2018)
- [7] Y. Wu, S. Hirakawa, U. H. Reimers and J. Whitaker, "Overview of Digital Television Development Worldwide," in Proceedings of the IEEE, vol. 94, no. 1, pp. 8-21, Jan. 2006, doi: 10.1109/JPROC.2005.861000.
- [8] H. Saito, N. Inamoto and S. Iwase, "Sports scene analysis and visualization from multiple-view video," 2004 IEEE International Conference on Multimedia and Expo(ICME) (IEEE Cat. No.04TH8763), 2004, pp. 1395-1398 Vol.2, doi: 10.1109/ICME.2004.1394492.
- [9] Xu, Ming, Orwell, James, and Jones, Graeme. "Tracking football players with multiple cameras" 5. 2909 - 2912 Vol. 5. 10.1109/ICIP.2004.1421721.
- [10] Tiago Soares da Costa, Maria Teresa Andrade, and Paula Viana. 2021. SmoothMV: Seamless Content Adaptation through Head Tracking Analysis and View Prediction. In Proceedings of the International Workshop on Immersive Mixed and Virtual Environment Systems (MMVE '21) (MMVE '21). Association for Computing Machinery, New York, NY, USA, 8-13. <https://doi.org/10.1145/3458307.3463814>
- [11] Intel. "Intel RealSense Technology". Available on: www.intel.com/content/www/us/en/architecture-and-technology/realsense-overview.html. Last access: 02/01/2023
- [12] FFMPEG.org. "FFMPEG 4.0 Wu". Available on: <https://ffmpeg.org/>. Last access: 03/01/2023.
- [13] GPAC Multimedia Open Source Project. "GPAC 1.0.0 Release Build". Available on: <https://gpac.wp.imt.fr/>. Last access: 03/01/2023.
- [14] Nuno Silva, Tiago Soares da Costa e Maria Teresa Andrade. "Prediction of Visual Behaviour in Immersive Contents", MsD Dissertation, Department of Electrical and Computer Engineering, Faculty of Engineering of the University of Porto. 2022.
- [15] A. Mondal, S. Sengupta, B. R. Reddy, M. J. V. Koundinya, C. Govindarajan, P. De, N. Ganguly, and S. Chakraborty, "Candid with YouTube," Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video, 2017.
- [16] M. Rozen and M. Wallace, "Adobe Flash Platform Achieves Record Adoption" Available on: <https://news.adobe.com/news/news-details/2009/Adobe-Flash-Platform-Achieves-Record-Adoption/default.aspx>. Last Access: 04/01/2023.

- [17] The Wowza Team, "Video Streaming Latency Report", Wowza Media Systems, Lakewood, Colorado USA, 2020. Available in: <https://www.wowza.com/wp-content/uploads/Streaming-Video-Latency-Report-Interactive-2020.pdf> Last Access: 04/01/2023
- [18] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," in IEEE MultiMedia, vol. 18, no. 4, pp. 62-67, April 2011, doi: 10.1109/M-MUL.2011.71.
- [19] Mueller, Christopher "MPEG-DASH (Dynamic Adaptive Streaming over HTTP)" Available on: <https://bitmovin.com/dynamic-adaptive-streaming-http-mpeg-dash>. Last Access: 04/01/2023
- [20] Ruether, Traci, "Video Codecs and Encoding: Everything You Should Know (Update)" Available on: <https://www.wowza.com/blog/video-codecs-encoding> Last Access: 05/01/2023
- [21] A. Yaqoob, T. Bi, G.M. Muntean. "A Survey on Adaptive 360° Video Streaming Solutions, Challenges and Opportunities". In IEEE Communications Surveys and Tutorials, doi: 10.1109/COM ST.2020.3006999.
- [22] H. d. J. O. Dominguez, O. O. V. Villegas, V. G. C. Sanchez, E. D. G. Casas and K. R. Rao, "The H.264 Video Coding Standard," in IEEE Potentials, vol. 33, no. 2, pp. 32-38, March-April 2014, doi: 10.1109/MPOT.2013.2284525.
- [23] G. J. Sullivan, J. -R. Ohm, W. -J. Han and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1649-1668, Dec. 2012, doi: 10.1109/TCSVT.2012.2221191.
- [24] J. Han et al., "A Technical Overview of AV1," in Proceedings of the IEEE, vol. 109, no. 9, pp. 1435-1462, Sept. 2021, doi: 10.1109/JPROC.2021.3058584.
- [25] A. Filippov and V. Ruffitskiy, "Recent Advances in Intra Prediction for the Emerging H.266/VVC Video Coding Standard," 2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON), 2019, pp. 0525-0530, doi: 10.1109/SIBIRCON48586.2019.8958416.
- [26] ISO. "ISO/IEC 23009-3:2021 - Information technology — Coded representation of immersive media – Part 3: Versatile video coding". Available on: <https://www.iso.org/standard/73022.html>. Last access: 11/03/2021.
- [27] A. Vetro, T. Wiegand, G.J. Sullivan. "Overview of the stereo and multiview video coding extensions of the H.264/MPEG-4 AVC standard". In Proceedings of the 2011 IEEE, vol. 99, 4, pp. 626-642
- [28] Bitmovin. "2021 Bitmovin Video Developer Report". Available on: <https://bitmovin.com/video-developer-report>. Last Access: 05/01/2023
- [29] BoxCast Team "HEVC (H.265) vs. AVC (H.264): What's the Difference?" Available on: "https://www.boxcast.com/blog/hevc-h.265-vs.-h.264-avc-whats-the-difference". Last Access: 05/01/2023
- [30] Wonderfox Team, "H.265 VS H.264 - The Significant Advantages of H.265 over H.264", Available on: <https://www.videoconverterfactory.com/tips/h265vsh264.html>. Last Access: 05/01/2023
- [31] Bruhanth Mallik and Akbar Sheikh Akbari. "Hevc Based Multi-view Video Codec Using Frame Interleaving Technique" In 2016 9th International Conference on Developments in eSystems Engineering (DeSE), pages 181–185, 2016.
- [32] Seif Allah El Mesloul Nasri. "Multiview Coding and Compression for 3D Video". PhD Thesis, Université Badji Mokhtar - Annaba (Algérie), July 2018
- [33] Shailendra Kumar, USMANI T, Syed Saeed, and Naimur Kidwai. "A Comparative Analysis of Ddvance Three Dimensional Video Coding for Mobile Three Dimensional TV".

International Journal of Electronics and Communication Engineering (IJECE) ISSN (E) 2278-991X, 3:59– 64, 09 2014

[34] D. Salcedo C. Guerrero and R. Martinez "Available Bandwidth Estimation Tools: Metrics, Approach and Performance", in International Journal of Communication Networks and Information Security, Vol. 10, No. 3, December 2018

[35] Mukta, M. and Gupta, N. (2017). Bandwidth Estimation Tools and Techniques: A Review. International Journal of Research, 4(13), 1250-1265

[36] R. Prasad and Others. "Bandwidth estimation: metrics, measurement techniques, and tools". In: IEEE Network 17.6 (2003), pp. 27–35.

[37] M. Mathis and M. Allman, "A Framework for Defining Empirical Bulk Transfer Capacity Metrics", RFC 3148.

[38] J. Strauss and M. Frans Kaashoek, "Estimating Bulk Transfer Capacity", MIT Laboratory for Computer Science.

[39] F. Pouzols "Comparative Analysis of Active Bandwidth Estimation Tools", Passive and Active Network Measurement, 5th International Workshop, PAM 2004, Antibes Juan-les-Pins, France, April 19-20, 2004

[40] T. Arsan, "Review of bandwidth estimation tools and application to bandwidth adaptive video streaming," High Capacity Optical Networks and Emerging/Enabling Technologies, 2012, pp. 152-156, doi:10.1109/HONET.2012.6421453.

[41] I. Dedinski, H. Meer, L. Han L., L. Mathy D. D. Pezaros, J. Sventek, X. Zhan, "Cross-Layer Peer-to-Peer Track Identification and Optimization Based on Active Networking" 13-27. 10.1007/978-3-642-00972-3.2.

[42] Vocal, "802.11 Distributed Coordination Function (DCF)", Available on: vocal.com/networking/802-11-distributed-coordination-function-dcf/ Last Access: 09/01/2023

[43] Giuseppe Binachi, Henia Tinnirello, "Remarks on IEEE 802.11 DCF performance Analysis", in IEEE Communications Letters, Aug. 2005, vol. 9, no. 8.

[44] R. Sitaraman and R. Barton, "Method and apparatus for measuring stream availability, quality and performance," US Patent 7,010,598, Feb. 2003.

[45] J. Dilley, B. M. Maggs, J. Parikh, H. Prokop, R. K. Sitaraman, and W. E. Weihl, "Globally distributed content delivery," IEEE Internet Comput., vol. 6, no. 5, pp. 50–58, Sep.–Oct. 2002

[46] S. S. Krishnan and R. K. Sitaraman, "Video Stream Quality Impacts Viewer Behavior: Inferring Causality Using Quasi-Experimental Designs," in IEEE/ACM Transactions on Networking, vol. 21, no. 6, pp. 2001-2014, Dec. 2013, doi: 10.1109/T-NET.2013.2281542.

[47] SSIMWAVE, "Is Live Sports Streaming Striking Out On Quality?". Available on: <https://blog.ssimwave.com/live-sports-streaming-striking-out-on-quality>. Last Access: 10/01/2023

[48] H. Chen, S. Ng, and A. Rao, "Cultural Differences in Consumer Impatience," J. Market. Res., vol. XLII, pp. 291–301, 2005.

[49] J. Harvey and S. Le Moan, "Gradual Chroma Reduction and High-Level Visual Masking in Videos," 2020 IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emirates, 2020, pp. 151-155, doi: 10.1109/ICIP40778.2020.9191069.

[50] T. Lyko, M. Broadbent, N. Race, M. Nilsson, P. Farrow and S. Appleby, "Llama - Low Latency Adaptive Media Algorithm," 2020 IEEE International Symposium on Multimedia (ISM), Naples, Italy, 2020, pp. 113-121, doi: 10.1109/ISM.2020.00027.

[51] G. Cermak, M. Pinson, and S. Wolf, "The relationship among video quality, screen resolution, and bit rate," IEEE Trans. Broadcast., vol. 57, no. 2, pp. 258–262, Jun. 2011.

- [52] C. Kreuzberger, D. Posch, and H. Hellwagner, "A scalable video coding dataset and toolchain for dynamic adaptive streaming over HTTP," in Proc. 6th ACM Multimedia Syst. Conf. (MMSys), 2015, pp. 213–218. [Online]. Available: <http://doi.acm.org/10.1145/2713168.2713193>
- [53] Z. Duanmu, K. Zeng, K. Ma, A. Rehman, and Z. Wang, "A quality of-experience index for streaming video," IEEE J. Sel. Topics Signal Process., vol. 11, no. 1, pp. 154–166, Feb. 2017.
- [54] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP," in IEEE Communications Surveys and Tutorials, vol. 21, no. 1, pp. 562–585, Firstquarter 2019, doi: 10.1109/COMST.2018.2862938.
- [55] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in Proc. IEEE INFOCOM 35th Annu. Int. Conf. Comput. Commun., Apr. 2016, pp. 1–9
- [56] Detti, B. Ricci, and N. Blefari-Melazzi, "Tracker-assisted rate adaptation for MPEG DASH live streaming," in Proc. IEEE INFOCOM 35th Annu. IEEE Int. Conf. Comput. Commun., Apr. 2016, pp. 1–9.
- [57] R. K. P. Mok, X. Luo, E. W. W. Chan, and R. K. C. Chang, "QDASH: A QoE-aware DASH system," in Proc. ACM 3rd Multimedia Syst. Conf. (MMSys), 2012, pp. 11–22. [Online]. Available: <http://doi.acm.org/10.1145/2155555.2155558>
- [58] R. Bourdon, WampServer, Available on: "<https://www.wampserver.com/en/>", Last Access: 14/01/2023
- [59] The Iperf Team, iPerf - The ultimate speed test tool for TCP, UDP and SCTP, Available on "<https://iperf.fr/>" Last Access: 14/01/2023
- [60] Oracle and/or its affiliates, "Chapter 6. Virtual Networking - Introduction to Networking Modes," in Oracle VM VirtualBox User Manual. Available in: www.virtualbox.org/manual/ch06.html Last Access: 16/01/2023
- [61] Microsoft Team, "Windows Subsystem for Linux Documentation "in Microsoft Technical Documentation. Available on: "<https://learn.microsoft.com/en-us/windows/wsl/>" Last Access: 16/01/2023
- [62] H. Liu, "Enhancements for Thrulay", Google Summer of Code Project, Final Project Report, Nov. 2015.
- [63] U.S Department of Veteran Affairs, "VA Technical Reference Model v 23.1" , Available on "<https://www.oit.va.gov/Services/TRM/ToolPage.aspx?tid=9304#>" Last Access: 16/01/2023
- [64] Dovrolis, Constantine Ramanathan, Parameswaran & Moore, D. (2005). "Packet-Dispersion Techniques and a Capacity-Estimation Methodology." in Networking, IEEE/ACM Transactions on. 12. 963 - 977. 10.1109/TNET.2004.838606.
- [65] J. Sommers, P. Barford and W. Willinger, "A Proposed Framework for Calibration of Available Bandwidth Estimation Tools," 11th IEEE Symposium on Computers and Communications (ISCC'06), Cagliari, Italy, 2006, pp. 709–718, doi: 10.1109/ISCC.2006.15.
- [66] Ningning Hu and P. Steenkiste, "Evaluation and Characterization of Available Bandwidth Probing Techniques," in IEEE Journal on Selected Areas in Communications, vol. 21, no. 6, pp. 879–894, Aug. 2003, doi: 10.1109/JSAC.2003.814505.
- [67] Ningning Hu and P. Steenkiste, "Estimating Available Bandwidth Using Packet Pair Probing", School of Computer Science Carnegie Mellon University, Pittsburgh, PA 15213, Sept, 2002.
- [68] Ningning Hu and P. Steenkiste, "IGI/PTR (Initial Gap Increasing / Packet Transmission Rate)" Available on: "<http://www.cs.cmu.edu/hnn/igi/>" Last Access: 16/01/2023

[69] Cottrell, Les. 2003. "pathChirp: Efficient Available Bandwidth Estimation for Network Paths". United States. <https://doi.org/10.2172/813038>. <https://www.osti.gov/servlets/purl/813038>.

[70] Emanuele Goldoni, Giuseppe Rossi and Alberto Torelli, "ASSOLO, a New Method for Available Bandwidth Estimation", In Proc. of The Fourth International Conference on Internet Monitoring and Protection (ICIMP 2009), Venice/Mestre, Italy, 24-28 May 2009 (ISBN 978-0-7695-3612-5

[71] IBM Watson Media, "Internet Connection and Recommended Encoding Settings" Available on: <https://support.video.ibm.com/hc/en-us/articles/207852117-Internet-connection-and-recommended-encoding-settings>. Last Access: 18/01/2023

[72] Canadian Space Agency, "Sleeping in Space", uploaded on Apr. 2013. Available on: <https://www.youtube.com/watch?v=UyFYgeE32f0&t=101s>. Last Access: 18/01/2023

[73] GoPro, "GoPro: Foggy Forest MTB", uploaded on Sept. 2019. Available on: <https://www.youtube.com/watch?v=MWyBgudQqsI>. Last Access: 18/01/2023

[74] The Official Saif, "Harry Kane Penalty Miss England Vs France World cup quarter finals", uploaded on Dec 2022. Available on: https://www.youtube.com/watch?v=J_4NMn9JjwM. Last Access: 18/01/2023

[75] NetLimiter, Locktime Software, 2005. Available on "<https://www.netlimiter.com/download>". Last Access: 19/01/2023

[76] Peter Christiansen, "How Much Speed Do I Need to Stream Video?", Nov 2022. Available on: "<https://www.highspeedinternet.com/resources/how-much-speed-do-i-need-to-watch-netflix-and-hulu>". Last Access: 19/01/2023

[77] Jordan Sheldrick, "Bandwidth for streaming: How much do you need?", Oct 2021. Available on: "<https://www.epiphan.com/blog/bandwidth-for-streaming>". Last Access: 19/01/2023