

# SLIGHT-DELAY SHAPED VARIABLE BIT RATE (SD-SVBR) TECHNIQUE FOR VIDEO TRANSMISSION

A thesis submitted to the Awang Had Salleh Graduate School of Arts and Sciences in  
full fulfillment of the requirements for the degree of Doctor of Philosophy Universiti  
Utara Malaysia

by

AHMAD SUKI BIN CHE MOHAMED ARIF

© 2011, AHMAD SUKI

# **PERMISSION TO USE**

In presenting this thesis in fulfillment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the University Library may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence by the Dean of Awang Had Salleh Graduate School. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my thesis.

Requests for permission to copy or to make other use of materials in this thesis, in whole or in part, should be addressed to

Dean of Awang Had Salleh Graduate School of Arts and Sciences

UUM College of Arts and Sciences

Universiti Utara Malaysia

06010 UUM Sintok

Kedah Darul Aman

# ABSTRAK

Matlamat utama tesis ini adalah bagi mempersembahkan satu bentuk baru Kadar Bit Pemboleh Ubah (VBR) untuk penghantaran video, yang memainkan peranan penting dalam penghantaran trafik video melalui Internet. Ini adalah kerana terdapat peningkatan mendadak dalam aplikasi media video melalui Internet, dan data video biasanya berciri trafik yang meletus secara mendadak, yang mengarah kepada ketidak-tetapan lebar jalur Internet. Algoritma baru pembentukan ini, yang dirujuk sebagai Sedikit Lengah - Pembentukan Kadar Bit Pemboleh Ubah (SD-SVBR), bertujuan untuk mengawal kadar kelajuan video untuk penghantaran aplikasi video. Ianya direka-bentuk berdasarkan algoritma Pembentukan Kadar Bit Pemboleh Ubah (SVBR) dan telah dilaksanakan dalam Penyelaku Rangkaian 2 (ns-2). Algoritma SVBR telah direka untuk aplikasi video masa nyata dan ia mewarisi beberapa keterbatasan dan kelemahan kerana pada proses-prosesnya terselit anggaran atau ramalan. SVBR mengalami beberapa masalah, seperti terjadinya penurunan mendadak kadar data yang tidak diingini, baldi limpah atas, kewujudan kadar data rendah, dan pengenerasian ketidak-tetapan negatif yang berulang. Algoritma baru ini mampu menghasilkan satu kadar data tinggi dan pada masa yang sama kestabilan Pengkuantuman Parameter (QP) yang lebih baik pada babak video. Sebagai tambahan, kadar data diperbentuk dengan cekap untuk mengelakkan kenaikan atau penurunan mendadak yang tidak diingini, dan untuk mengelakkan baldi limpah atas. Bagi mencapai matlamat tersebut, SD-SVBR memiliki tiga strategi; memproses Kumpulan Gambar (GoP) hadapan babak video dan mendapatkan senarai QP-ke-kadar data, mendimensikan kadar data kepada penggunaan tinggi baldi-bocor, dan melaksanakan langkah secara berhati-hati dalam mengikuti nilai QP sebelumnya. Walau bagaimanapun, algoritma ini perlu disekalikan dengan algoritma yang dapat memberi maklum balas status rangkaian untuk memberi kebaikan kawalan kadar video yang menyeluruh. Sebuah kombinasi beberapa klip video yang terdiri daripada berbagai kadar data video, telah digunakan untuk tujuan menilai prestasi SD-SVBR. Keputusan kajian menunjukkan bahawa SD-SVBR berjaya memperolehi nilai Nisbah Puncak Isyarat-kepada-Hingar (PSNR) yang hebat secara menyeluruh. Dalam pada itu, pada hampir kesemua kes, ia memperolehi satu kadar data tinggi tetapi tanpa baldi limpah atas, memanfaatkan penggunaan baldi dengan baik, dan yang menariknya, ia masih berupaya memperolehi satu ketidak-tetapan QP yang lebih rata.

# ABSTRACT

The aim of this thesis is to present a new shaped Variable Bit Rate (VBR) for video transmission, which plays a crucial role in delivering video traffic over the Internet. This is due to the surge of video media applications over the Internet and the video typically has the characteristic of a highly bursty traffic, which leads to the Internet bandwidth fluctuation. This new shaped algorithm, referred to as Slight Delay - Shaped Variable Bit Rate (SD-SVBR), is aimed at controlling the video rate for video application transmission. It is designed based on the Shaped VBR (SVBR) algorithm and was implemented in the Network Simulator 2 (ns-2). SVBR algorithm is devised for real-time video applications and it has several limitations and weaknesses due to its embedded estimation or prediction processes. SVBR faces several problems, such as the occurrence of unwanted sharp decrease in data rate, buffer overflow, the existence of a low data rate, and the generation of a cyclical negative fluctuation. The new algorithm is capable of producing a high data rate and at the same time a better quantization parameter (QP) stability video sequence. In addition, the data rate is shaped efficiently to prevent unwanted sharp increment or decrement, and to avoid buffer overflow. To achieve the aim, SD-SVBR has three strategies, which are processing the next Group of Picture (GoP) video sequence and obtaining the QP-to-data rate list, dimensioning the data rate to a higher utilization of the leaky-bucket, and implementing a QP smoothing method by carefully measuring the effects of following the previous QP value. However, this algorithm has to be combined with a network feedback algorithm to produce a better overall video rate control. A combination of several video clips, which consisted of a varied video rate, has been used for the purpose of evaluating SD-SVBR performance. The results showed that SD-SVBR gains an impressive overall Peak Signal-to-Noise Ratio (PSNR) value. In addition, in almost all cases, it gains a high video rate but without buffer overflow, utilizes the buffer well, and interestingly, it is still able to obtain smoother QP fluctuation.

# DECLARATION

Some of the work presented in this thesis have been published as listed below.

[1] A. Suki M. Arif, Osman Ghazali and Suhaidi Hassan, A Survey on Buffer and Rate Adaptation Optimization in TCP-Based Streaming Media Studies, International Conference on Network Applications, Protocols and Services 2008 (NetApps2008), ISBN 978-983-2078-33-3, on 21 - 22 Nov. 2008. [This paper has appeared in IEEE Xplore. Indexed by Scopus and IEEE Xplore]

[2] A. Suki M. Arif, Suhaidi Hassan, Osman Ghazali and Shahrudin Awang Nor. "Evalvid-RASV: Shaped VBR Rate Adaptation Stored Video System," in the Proceedings of *The 2010 International Conference on Information and Network Technology (ICINT 2010)*, Shanghai, China, vol. 5, pp. 246-250, Jun 2010. [This paper has been included in the IEEE Xplore and CSDL, and submitted to INSPEC, Thomson ISI Proceeding (ISTP), Ei Compendex for indexing.]

[3] Shahrudin Awang Nor, Suhaidi Hassan, Osman Ghazali and A. Suki M. Arif. "Friendliness of DCCP towards TCP over Large Delay Link Network," in the Proceedings of *The 2010 International Conference on Information and Network Technology (ICINT 2010)*, Shanghai, China, vol. 5, pp. 286-291, Jun 2010. [This paper has been included in the IEEE Xplore and CSDL, and submitted to INSPEC, Thomson ISI Proceeding (ISTP), Ei Compendex for indexing.]

[4] Arif, A.S.M., Hassan, S., Ghazali, O. & Nor, S.A. The Relationship of TFRC Congestion Control to Video Rate Control Optimization, International Conference on Network Applications, Protocols and Services (netApps2010), IEEE Computer Society, 2010, pp. 31-36. [This paper has appeared in IEEE Xplore. Indexed by Scopus and IEEE Xplore]

[5] Nor, S.A., Hassan, S., Ghazali, O., & Arif, A.S.M. On the Performance of TCP Pacing with DCCP, International Conference on Network Applications, Protocols and Services (netApps2010), IEEE Computer Society, 2010, pp. 37-41. [This paper has appeared in IEEE Xplore. Indexed by Scopus and IEEE Xplore]

[6] A. Suki M. Arif, Suhaidi Hassan, Osman Ghazali and Shahrudin Awang Nor. “VBR vs CBR: The Shaped VBR Stored Video Evalvid-RASV Mechanism is the Winner!,” in the Proceedings of *the 5th Social Economic and Information Technology (SEiT)*, Hatyai, Thailand. November 2010.

[7] A. Suki M. Arif, Suhaidi Hassan, Osman Ghazali, and Mohammed M. Kadhum, "Enhancing Shaped VBR Rate Control Algorithm for Stored Video Transmission System," in the Proceeding of The 2010 International Conference on Modeling, Simulation and Control, ICMSC 2010, Cairo, Egypt, pp. 205-209, 2010. [This paper has been included in the IEEE Xplore, and indexed by the Ei Compendex and Thomson ISI (ISTP)]

[8] A. Suki M. Arif, Suhaidi Hassan, Osman Ghazali, Mohammed M. Kadhum “Empirical Evaluation of the Shaped Variable Bit Rate Algorithm for Video Transmission,” International Journal of Computer Science and Information Security (IJCSIS), Vol. 9, No. 3, March 2011. [IJCSIS Publications Indexed @ [Google Scholar] @ [SCIRUS ] @ [ScientificCommons] @ [DOAJ]. ***This paper has been categorized as “The Best Paper”***]

[9] A. Suki M. Arif, Suhaidi Hassan, Osman Ghazali, Mohammed M. Kadhum “Design of a New Shaped Control Algorithm for a Video Application,” Accepted to International Journal of Digital Content Technology and its Applications (JDCTA), 2011. [JDCTA Publications Indexed @ ISI Thomson(under review), Elsevier, EI(Confirmed), SCOPUS(Confirmed), INSPEC(confirmed), PROQUESTS(confirmed) and many other citation databases]. ***This paper has received encouraging comments, “The work is solid and comprehensive. Overall presentation of this paper is good.”***]

# ACKNOWLEDGEMENT

In the name of ALLAH, The Most Gracious, and The Most Merciful.

Although PhD work is a lonely journey of individual endeavor, this work would not have come to fruition without the support of many people, to whom I am sincerely indebted and thankful.

My deepest gratitude is dedicated to my supervisors, Assoc. Prof. Dr. Suhaidi Hassan and Dr. Osman Ghazali, including my informal supervisor Dr Mohamad M. Kadhun, for their continuous guidance, fruitful feedback, moral support, and sharing of all their research experiences throughout these challenging years. They have eagerly provided a surplus of advices and constructive comments as well as optimism and encouragement at times when things were not looking rosy. Their detailed and constructive comments have helped me to better shape my research ideas.

Besides them, my gratitude to all my colleagues in the PhD journey during the monthly research camps and other colleagues; among them are Dr Massudi Mahmud, Dr Omar Almomani, Dr Angela Ampawan, Muhammad Shakirin, Yaser Miaji, Hasbullah, Ahmad Hanis, Shahrudin, and many others, specifically for the discussions and sometimes the heated arguments on the better ways to perform research, to construct the research objectives and title, etc. They were not only contributing constructive ideas on my research work, but some of them have also read parts of my thesis. I am also very thankful to Dr Arni Lie from SINTEF ICT, Dept. of Communication Systems, Trondheim, Norway, for providing valuable information and suggestions on issues related to video quality research and eagerly responding to my questions.

Finally, special thanks for my family, especially my beloved wife, for her patience and support throughout my three years plus of difficult endeavor. I guess they are the most who suffered throughout this period.

For Allah, *Alhamdulillah*. For others, *Jazakumullahu khairan katsiraan*.

# TABLE OF CONTENTS

<b>PERMISSION TO USE</b>	<b>i</b>
<b>ABSTRAK</b>	<b>ii</b>
<b>ABSTRACT</b>	<b>iii</b>
<b>DECLARATION</b>	<b>iv</b>
<b>ACKNOWLEDGEMENT</b>	<b>vi</b>
<b>TABLE OF CONTENTS</b>	<b>vii</b>
<b>LIST OF TABLES</b>	<b>xiii</b>
<b>LIST OF FIGURES</b>	<b>xiv</b>
<b>ABBREVIATIONS</b>	<b>xvii</b>
<b>CHAPTER ONE: INTRODUCTION</b>	
1.1 Video Transmission and Rate Control Issues . . . . .	1
1.1.1 Video Rate Control . . . . .	4
1.1.2 Video Transmission Rate Control Schemes . . . . .	6
1.1.3 Video Performance Evaluation . . . . .	7
1.2 Research Problem . . . . .	8
1.3 Research Question . . . . .	9
1.4 Research Motivation . . . . .	9
1.5 Scope of Research . . . . .	10
1.6 Research Objectives . . . . .	11
1.7 Key Research Steps . . . . .	12
1.8 Key Contributions . . . . .	12



1.9	Organization of the Thesis . . . . .	13
-----	--------------------------------------	----

## CHAPTER TWO: BACKGROUND AND RELATED WORK

2.1	The Challenges in Transporting Video over the Internet . . . . .	16
2.2	Video and Coding . . . . .	17
2.2.1	Video Sequence . . . . .	18
2.2.2	Video Hierarchy . . . . .	19
2.2.2.1	Video Sequence and GoP . . . . .	19
2.2.2.2	Frame, Frame Type, and the Relationship with GoP . . . . .	20
2.2.2.3	Block, Macroblock, and Slice . . . . .	21
2.2.3	Video Coding . . . . .	21
2.2.3.1	Inter- and Intra-Frame Coding . . . . .	22
2.2.3.2	Video Coding Techniques . . . . .	23
2.2.3.3	Advancement in Encoding Techniques and Standards . . . . .	24
2.3	Video Rate Control . . . . .	28
2.3.1	Rate Control and Video Coding . . . . .	29
2.3.2	Understanding the Relationship between Quantization Parameter (QP) and the Rate Control . . . . .	31
2.3.3	Rate-Distortion Theory . . . . .	33
2.3.4	Traditional Rate Controllers: VBR versus CBR . . . . .	34
2.3.5	Rate Control Research . . . . .	36
2.3.5.1	R-D Modelling . . . . .	37
2.3.5.2	Quality Constrained Rate Control . . . . .	39
2.3.5.3	From Objective Optimization to Subjective Optimization . . . . .	40
2.3.5.4	Hybrid Structure-Based Optimization . . . . .	41
2.4	Network Transmission Protocol . . . . .	41
2.4.1	TCP-Based Video Transmission . . . . .	42
2.4.2	TCP with Adaptive Bit Rate Transmission . . . . .	43
2.4.3	UDP-Based Video Transmission Research . . . . .	44
2.4.4	Alternative Protocol for Video Application . . . . .	45
2.5	Related Work on Video Rate Shaping Algorithm . . . . .	47
2.5.1	Leaky-Bucket Algorithm . . . . .	48
2.5.2	Video Rate Shaping Research in Stored-Video Application . . . . .	49
2.5.3	Video Rate Shaping Research in Real-Time Video Application . . . . .	51
2.5.4	Recent Work on Video Rate Shaping . . . . .	52

2.5.5	Shaped Variable Bit Rate (SVBR) . . . . .	54
2.5.5.1	Shaped Variable Bit Rate (SVBR) . . . . .	54
2.5.5.2	Recent SVBR Related Work . . . . .	55
2.6	Video Performance Evaluation . . . . .	58
2.6.1	Video Performance Evaluation Categorization . . . . .	60
2.6.2	PSNR . . . . .	63
2.6.3	Evalvid . . . . .	64
2.6.4	Evalvid-ns2 Integration and Evalvid-RA . . . . .	65
2.7	Summary . . . . .	67

### **CHAPTER THREE: RESEARCH METHODOLOGY**

3.1	Research Operational Framework . . . . .	69
3.2	Methods used in Evaluating Network Performance . . . . .	73
3.2.1	Network Performance Evaluation Approach Consideration . . . . .	75
3.2.2	Justification for Simulation Method . . . . .	78
3.2.3	Network Simulation 2 (ns2) . . . . .	81
3.3	Evaluation Metrics . . . . .	83
3.3.1	PSNR . . . . .	84
3.3.2	Video Rate . . . . .	84
3.3.3	QP Stability . . . . .	84
3.4	Experimental Setting . . . . .	86
3.4.1	Video Traffic . . . . .	86
3.4.2	Video Sequences Used . . . . .	89
3.4.3	Network Simulation Setup . . . . .	89
3.4.3.1	The Transmission Protocol . . . . .	90
3.4.3.2	Network Simulation Topology . . . . .	91
3.4.3.3	Network Simulation Settings . . . . .	92
3.5	Validation and Verification . . . . .	93
3.5.1	Validation of ns-2 . . . . .	94
3.5.2	Validation of the Simulation . . . . .	94
3.5.3	The New Algorithm Implementation Verification and Validation . . . . .	95
3.6	Summary . . . . .	95

### **CHAPTER FOUR: EMPIRICAL EVALUATION OF THE SHAPED VARIABLE BIT RATE (SVBR) ALGORITHM**

4.1	Exploiting the Advantages of CBR and VBR . . . . .	98
4.2	SVBR Algorithm . . . . .	99

4.2.1	SVBR Principles . . . . .	99
4.2.1.1	Controlling Video Data Input . . . . .	99
4.2.1.2	Allowing VBR Coding when the Network Permits . . . . .	101
4.2.2	Determining Suitable Bit Rate Allocation and Quantization Parameter Values . . . . .	103
4.2.2.1	Determining Bit Rate Allocation . . . . .	104
4.2.2.2	Determining QP Value . . . . .	105
4.2.2.3	The Implications of the Design . . . . .	108
4.2.3	SVBR Algorithm Flow . . . . .	109
4.3	Evaluation of Strengths and Weaknesses of SVBR . . . . .	111
4.3.1	The Evaluation Approach . . . . .	112
4.3.2	Rate Control Experiment Settings . . . . .	112
4.3.3	The Result . . . . .	112
4.3.3.1	Observation on the SVBR video rate and Bucket Fullness Level . . . . .	113
4.3.3.2	Observation on the PSNR and QP Values . . . . .	114
4.3.4	The Analysis . . . . .	117
4.4	Summary . . . . .	133

## **CHAPTER FIVE: DESIGN AND IMPLEMENTATION OF A NEW SHAPED CONTROL ALGORITHM FOR A VIDEO APPLICATION**

5.1	Design Objectives . . . . .	135
5.2	SD-SVBR Algorithm . . . . .	136
5.2.1	Avoiding Unwanted video rate Increment/Decrement by Using Real QP-to-video rate Matching . . . . .	136
5.2.1.1	Justification for Producing Correct QP and video rate Value . . . . .	138
5.2.1.2	Obtaining Correct Corresponding QP Value . . . . .	138
5.2.1.3	The Benefit Gained . . . . .	141
5.2.2	Increasing the video rate . . . . .	145
5.2.2.1	Design a Higher video rate Video Controller . . . . .	147
5.2.2.2	The High video rate Algorithm . . . . .	149
5.2.2.3	High video rate Algorithm Achievement . . . . .	151
5.2.3	Smoothing the Fluctuation . . . . .	157
5.2.3.1	The Smoothing QP Value Principle . . . . .	159
5.2.3.2	Smoothing QP Value Algorithm . . . . .	165

5.2.3.3	Effects of the Smoothing QP Value . . . . .	168
5.3	SD-SVBR Implementation . . . . .	172
5.3.1	Work Environment Architecture . . . . .	172
5.3.2	Structure of Implementation: Finite State Machine . . . . .	175
5.3.3	Implementing the Algorithm . . . . .	177
5.3.3.1	Main Algorithm Flow . . . . .	178
5.3.3.2	Tcl Program Flow . . . . .	178
5.3.3.3	C++ Program Flow . . . . .	182
5.3.4	Allowing Dynamic GoP Size . . . . .	185
5.4	Summary . . . . .	186

## CHAPTER SIX: RESULTS AND DISCUSSIONS

6.1	The Overall Result . . . . .	187
6.1.1	The PSNR Result . . . . .	189
6.1.2	The Video Rate Result . . . . .	191
6.1.3	Bucket Utilization Result . . . . .	193
6.1.3.1	Bucket Utilization . . . . .	193
6.1.3.2	Bucket Utilization in the Overall Perspective . . . . .	197
6.1.4	The QP Value Result . . . . .	199
6.2	Examining of the PSNR Result in Detail . . . . .	199
6.2.1	Analysis: Part I of the PSNR Result . . . . .	203
6.2.2	Analysis: Part II of the PSNR Result . . . . .	206
6.2.3	Analysis: Part III of the PSNR Result . . . . .	211
6.2.4	Analysis: Part IV of the PSNR Result . . . . .	217
6.2.5	Analysis: Part V of the PSNR Result . . . . .	221
6.3	Summary . . . . .	224

## CHAPTER SEVEN: CONCLUSION AND FUTURE RESEARCH WORK

7.1	Summary of the Research . . . . .	227
7.2	Contributions . . . . .	230
7.2.1	Empirical Evaluation of the SVBR Algorithm . . . . .	230
7.2.2	A New Shaped VBR Algorithm . . . . .	231
7.2.3	The implementation of the SD-SVBR . . . . .	233
7.3	Suggestions for Future Work . . . . .	236
7.3.1	Integrated Rate Control . . . . .	236
7.3.2	Advanced Codec Support . . . . .	238

7.3.3 Exploiting the Algorithm for Stored Video and/or Frame/Object Level . . . . .	239
<b>REFERENCES</b>	<b>240</b>
<b>Appendix A</b>	<b>257</b>
<b>Appendix B</b>	<b>262</b>
<b>Appendix C</b>	<b>269</b>

# LIST OF TABLES

2.1	Video coding standards . . . . .	22
2.2	QP values and GoP video rate . . . . .	32
3.1	Criteria for selecting performance evaluation technique (adapted from [1]) . . . . .	78
3.2	Profile of video sequences used in the experiments . . . . .	90
4.1	Relationship of bucket fullness, active sequence, and SVBR video rate	105
4.2	Relationship of bucket fullness level, active sequence, and QP value .	109
4.3	Examples of the relationship of bucket fullness level, active sequence, and QP value . . . . .	110
4.4	Data size for GoP 26-40 . . . . .	116
4.5	Respective QP values for GoP 198-204 . . . . .	120
4.6	Respective QP and video rate values for GoP-202 . . . . .	122
5.1	List of corresponding QP-to-video rate sample . . . . .	140
5.2	QP-to-video rate for SVBR and SD-SVBR at GoPs 200-202 . . . . .	143
5.3	QP-to-video rate for SVBR and SD-SVBR at GoPs 368-370 . . . . .	145
6.1	The overall statistical result . . . . .	188
6.2	Recalculation example for GoP 485-489 . . . . .	196
6.3	Recalculation example for GoP 27-34 . . . . .	197
6.4	A sample of the frame to GoP mapping table . . . . .	202

# LIST OF FIGURES

1.1	Video transmission application architecture . . . . .	2
1.2	Obvious mismatch between video rate and transport protocol rate . . .	3
1.3	General rate control architecture . . . . .	4
1.4	Three rate control schemes for the video transmission application . .	7
1.5	Scope of the research - video coding rate control only . . . . .	10
2.1	The GoP arrangement of frames type used in this research . . . . .	21
2.2	Relationship between rate controller and video coding . . . . .	30
2.3	video rate comparison for CBR and VBR . . . . .	34
2.4	Quantization Parameter comparison for CBR and VBR . . . . .	35
2.5	VBR and CBR setup . . . . .	35
2.6	Parameters in a leaky-bucket system . . . . .	50
2.7	The original Evalvid framework . . . . .	65
2.8	Evalvid framework in simulated environment . . . . .	66
2.9	Evalvid-RA framework (redraws from [2]) . . . . .	67
3.1	Research steps . . . . .	71
3.2	Performance evaluation approaches (adapted from [3]) . . . . .	73
3.3	Sample output of “psnr.exe” . . . . .	85
3.4	A typical video trace . . . . .	88
3.5	Simulation parameter setting . . . . .	92
4.1	Restrict input in the leaky-bucket algorithm . . . . .	100
4.2	SVBR shaping principle when bucket empty . . . . .	102
4.3	SVBR shaping principle when bucket full . . . . .	103
4.4	Rate control block diagram with SVBR . . . . .	106
4.5	SVBR algorithm flow chart . . . . .	111
4.6	The SVBR experiments - comparing the result with VBR and CBR . .	115
4.7	The SVBR experiments - bucket fullness level . . . . .	115

4.8	GoP data size calculation for TFRC transmission. . . . .	116
4.9	The SVBR experiments - PSNR value . . . . .	118
4.10	The SVBR experiments - QP values . . . . .	119
4.11	Emulating CBR with GoP video rate granularity . . . . .	119
4.12	Sharp decrease scenario in the GoP-201 . . . . .	121
4.13	Sharp VBR increase scenario . . . . .	124
4.14	A long-low-flat video rate sequence . . . . .	126
4.15	Positive fluctuation . . . . .	128
4.16	Negative fluctuation . . . . .	129
4.17	SVBR fluctuation . . . . .	130
4.18	Negative SVBR fluctuation with VBR lower than CBR . . . . .	132
4.19	SVBR with varied VBR video rate . . . . .	133
4.20	Associate QP values for the varied VBR video rate . . . . .	134
5.1	Working differences between SVBR and SD-SVBR algorithm . . . . .	137
5.2	Problem of generating the correct corresponding QP value . . . . .	139
5.3	Getting information of GoP-k+1 video sequence while processing suitable QP value for GoP=k+1 . . . . .	140
5.4	A flowchart for a QP selection . . . . .	142
5.5	QP-to-video rate result for SVBR and SD-SVBR at GoP 200-202 . . .	144
5.6	QP-to-video rate for SVBR and SD-SVBR at GoP 368-370 . . . . .	146
5.7	video rate categorization . . . . .	148
5.8	Sample C++ codes for the increased video rate algorithm . . . . .	149
5.9	SVBR at GoPs 2-6: Low rate trend . . . . .	152
5.10	SD-SVBR at GoPs 2-6: Breaking the SVBR limitation . . . . .	153
5.11	SD-SVBR at low SVBR bucket utilization . . . . .	154
5.12	SVBR at GoPs 368-371: Overflow bucket trend . . . . .	155
5.13	SVBR at GoPs 368-371: The real overflow . . . . .	156
5.14	SD-SVBR at GoPs 368-371: Maintaining the bucket level below $b$ . .	157
5.15	The effect of following a smaller QP value . . . . .	161
5.16	The effect of following a bigger QP value . . . . .	162
5.17	Sample C++ codes for the QP smoothing algorithm . . . . .	166
5.18	Smoothing QP values at GoPs 25-31 . . . . .	168
5.19	Corresponding video rate for SVBR algorithm at GoPs 25-31 . . . . .	169
5.20	Corresponding video rate for SD-SVBR algorithm at GoPs 25-31 . . .	170
5.21	A longer SD-SVBR QP constant at GoP 201-253 . . . . .	170



5.22	Smoothing and non-smoothing SD-SVBR QP value for GoP 201-253	171
5.23	The effect of QP smoothing approach at Gop 1 and 2	171
5.24	The effect on the video rate of QP smoothing approach at GoP1 and 2	173
5.25	Evalvid-RA environment parts	174
5.26	Finite State Machine Model of SD-SVBR Mechanism	176
5.27	The SD-SVBR main algorithm flow	179
6.1	The overall PSNR result	190
6.2	The overall video rate result	192
6.3	The overall bucket utilization result	194
6.4	The real bucket/buffer utilization	195
6.5	The “real” overall bucket utilization	198
6.6	The overall QP value result	200
6.7	Part I: PSNR result for frames 1 to 276	203
6.8	video rate, buffer utilization, and QP value for sub-part 1 of Part I	204
6.9	video rate, buffer utilization, and QP value for sub-part 2 of Part I	205
6.10	Part II: PSNR result for frames 325-2360	206
6.11	The PSNR sub-parts for the Part II chart	207
6.12	The video rate and QP value for sub-part 1 of Part II chart	209
6.13	The video rate and QP value for sub-part 2 of Part II chart	210
6.14	The video rate and QP value for sub-part 3 of Part II chart	211
6.15	Part III: PSNR result for frames 2439-3569	212
6.16	The PSNR sub-parts for the Part III chart	212
6.17	The video rate and QP value for sub-part 1 of Part III chart	214
6.18	The video rate and QP value for sub-part 2 of Part III chart	215
6.19	Part IV: PSNR result for frames 3612-4551	217
6.20	The PSNR sub-parts for the Part IV chart	218
6.21	The video rate and QP value for sub-part 1 of Part IV chart	219
6.22	QP before and after GoPs 303-317	220
6.23	The video rate and QP value for sub-part 2 of Part IV chart	221
6.24	Actual “bucket” utilization for GoP 366-379	222
6.25	Part V: PSNR result for frames 4554-6499	222
6.26	The PSNR sub-part for the Part V chart - frames	223
6.27	The video rate and QP value for sub-part of Part V chart	224
6.28	Higher performance gain before GoPs 464-492	225
6.29	Higher performance gain after GoPs 464-492	226

# ABBREVIATIONS

AQM	Active Queue Management
ASMP	Adaptive Smooth Multicast Protocol
ASSP	Adaptive Smooth Simulcast Protocol
AVC	Advanced Video Coding
BDP	Bandwidth Delay Product
B-frame	Bi-directional-frames
BL	base layer
bps	bits per second
C	chroma/chrominance
CBR	Constant Bit Rate
CIF	Common Intermediate Format
CZD	Czenakowski distance
DCCP	Datagram Congestion Control Protocol
DCT	Discrete Cosine Transform
DRS	Dynamic Rate Shaping
ECN	Explicit Congestion Notification
EL	enhancement layer
FGS	Fine Granular Scalability

fps	frame per seconds
GCRA	Generic Cell Rate Algorithm
GloMoSim	Global Mobile Simulator
GoP	Group of Picture
GS	Guaranteed Service
GUI	Graphical User Interface
HoD	Histogram of Difference
IEC	International Electrotechnical Commission
I-frame	Intra-frames
IFTP	Internet Friendly Transport Protocol
ISO	International Organization for Standardization
ITU-T	Telecommunication Standardization Sector of the International Telecommunications Union
JVT	Joint Video Team
LDA+	Loss-Delay Based Adaptation Algorithm
LPR	Linear Proportional Response
MB	Macroblock
MDC	Multiple Description Coding
MOS	Mean Opinion Score
MPEG	Moving Picture Experts Group
NAM	Network Animator
NCA	Nominee-Based Congestion Avoidance
NF	Neural-Fuzzy
ns-2	Network Simulator 2

NTSC	National Television Standards Committee
PAL	Phase Alternating Line
PEVQ	Perceptual Evaluation of Video Quality
P-frame	Predictive-frames
PSNR	Peak Signal-to-Noise Ratio
QCIF	Quarter Common Intermediate Format
QoS	Quality of Service
QP	Quantization Parameter
RAP	Rate Adaptation Protocol
RBF	Rule-Based Fuzzy
R-D	Rate-Distortion
REAL	Realistic And Large
RLA	Random Listening Algorithm
RR	Receiver Report
RTCP	Real-Time Transport Control Protocol
RTP	Real-time Transport Protocol
RTT	Round-Trip Time
SD-SVBR	Slight Delay - Shaped Variable Bit Rate
SNR	Signal-to-Noise Ratio
SR	Sender Report
SVBR	Shaped Variable Bit Rate
SVC	Scalable Video Coding
Tcl	Tool Command Language
TCP	Transmission Control Protocol

TEAR	TCP Emulation At Receiver
TES	Transform Expand Sample
TFRC	TCP-Friendly Rate Control
UDP	User Datagram Protocol
VBR	Variable Bit Rate
VBV	video buffer verifier
VoD	Video on Demand
VQM	Video Quality Measurement
Y	luma/luminance

# **CHAPTER ONE**

## **INTRODUCTION**

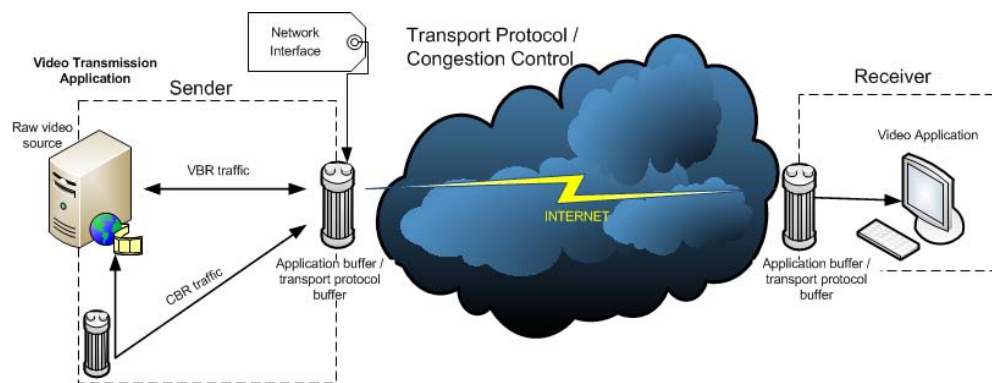
This thesis is about enhancing rate control or rate adaptation for video transmission, which plays an important role in delivering video traffic over the Internet. This is due to the fact that video data is highly bursty and the Internet bandwidth fluctuates. The aim of this chapter is to place the thesis in its context by initially covering a brief introduction to video rate control issues. Later, the description of the research properties are stated, which includes the motivation, scope, objectives, key research steps, and the contributions of the work done in this thesis. In the last section, the whole thesis organization is presented.

### **1.1 Video Transmission and Rate Control Issues**

Recent years have witnessed an explosive growth of the Internet and increasing demand for multimedia information services. Multimedia based applications via the Internet have received tremendous attention. The surge of video media applications over the Internet are attributed to the increasing capacity of the Internet and its cost-effectiveness. In spite of the growing networking capabilities of the modern Internet and sophisticated techniques used by today's video coding, transmitting video over the Internet is still a great challenging task, as stated in [4]. In order

to gain good quality video transmission application, several weaknesses need to be addressed. From the transport layer protocol perspective, the inherited problems are the lack of throughput guarantees, variabilities in bit rates, delays, and jitters. Those characteristics are not “friendly” to video applications because video applications favor timeliness to reliability. That means, video applications are able to compromise packet loss but sensitive to packet delay.

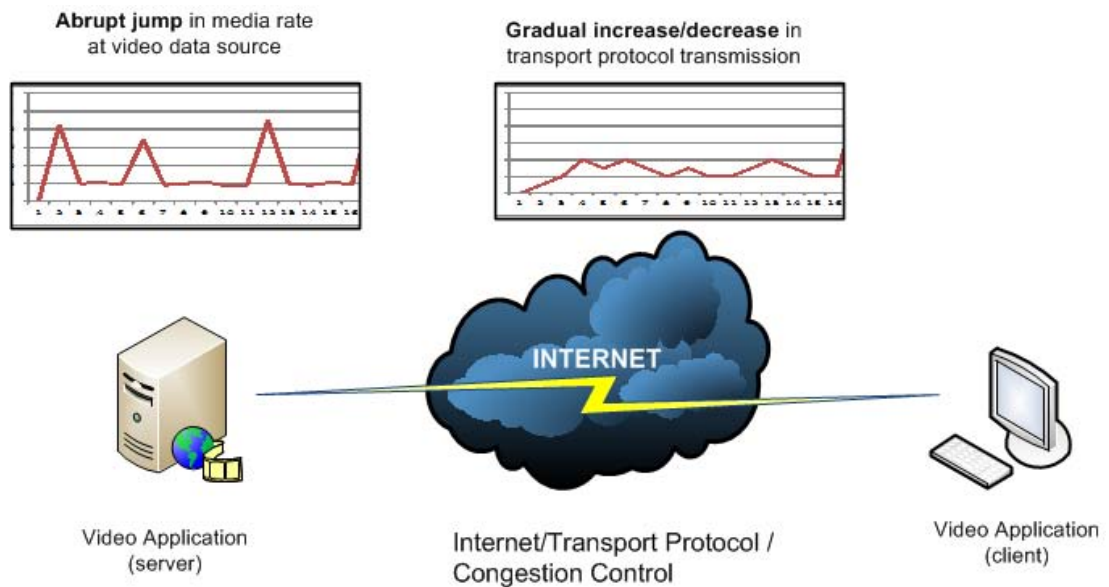
Figure 1.1 shows the relationship between video transmission application and transport protocol. There are two main activities in the diagram; the video coding process at the sender and the network transmission or congestion control at the Internet network. The sender performs video coding where raw video will be coded or compressed for transmission. The popular coded video rate controllers are Variable Bit Rate (VBR) and Constant Bit Rate (CBR). The coded video then is sent to the application buffer or sender’s transport protocol buffer for the transmission through the Internet. The transmission regulation is performed by the transport layer protocol and the real transmission will be done by the Internet. This protocol regulates congestion control, rate control, flow control, and other controls. of the application data.



**Figure 1.1:** Video transmission application architecture

However, as stated earlier, various real-time applications are becoming increasingly common, that means the Internet infrastructure must be able to accommodate those applications. Real-time applications have strict requirements in terms of throughput

guarantees, consistent bit rates, and fewer delays or jitters. In contrary, congestion control will regulate transmission rate without considering real-time application requirements. Thus, it is not surprising that Guo L. et al. [5] found that up to 40% media streaming users are suffering various quality degradations. It will become worse if there are regular occurrences of abrupt changes in video data. These abrupt changes will trigger bursty network traffic, uneven resource utilization, and may lead to congestion and exacerbate display disruptions [6]. Figure 1.2 illustrates an obvious mismatch between media rate and transport protocol transmission rate.



**Figure 1.2:** Obvious mismatch between video rate and transport protocol rate

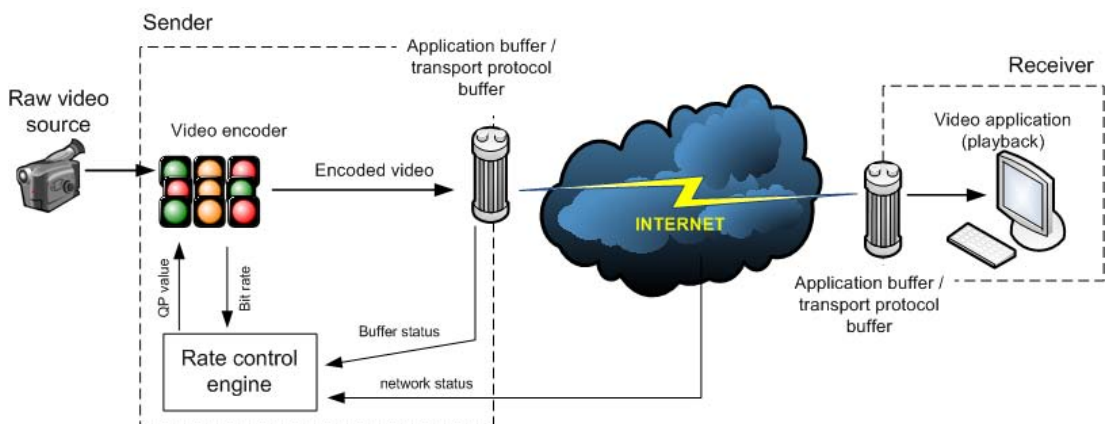
As a consequence of explosive growth of media traffic traversing the Internet in recent years, many optimization studies have been undertaken. These are due to the fact that media traffic data produces irregular traffic to the network, which is not designed to fulfill the requirements of such traffic natively (if certain adaptations to the Internet system are not perform). This irregular traffic has been discussed in previous works, such as [7]. Thus, the traffic will probably congest the network, and in the worst case scenario, it may lead to a congestion-collapsed network.



### 1.1.1 Video Rate Control

Having discussed the growing number of video-based applications over the Internet and the problems posed by this kind of application, it can be concluded that the admission of video traffic to the Internet needs to be regulated. If video traffic is not regulated, the traffic might eventually result in network congestion or data loss. Additionally, the traffic regulation opens to the possibility of the introduction of smoother-higher quality video data. The mechanism to regulate video traffic is called video rate control or rate adaptation.

Rate control is always regarded as an essential element of the typical video coding as stated in previous works [8, 9, 10, 11, 12, 13, 14]. The general rate control is illustrated in Figure 1.3, where its main task is to regulate the coded video bits to meet a suitable target rate. Video coding refers to the process of reducing the quantity of data used to represent a sequence of video pictures or frames. A number of standards of video coding have been defined, such as MPEG-2, H.263, and MPEG-4. Although rate control is not always a part of video coding, it plays a very important role in producing video data traffic [15, 16, 17, 18]. Since it is not a fixed part of the video coding, this leaves the flexibility to designers to develop a suitable scheme for specific applications [14]. Thus, any created algorithm can be applied to many video coding standards.



**Figure 1.3:** General rate control architecture

The task of the rate controller module is complicated by the fact that the encoded video quality should be kept at the highest possible quality level for each frame and within each frame, while avoiding such visual artifacts as blurring, blocking, and jitter. The goal of rate control, therefore, is to keep the output bit rate within constrained limits such that the buffer does not overflow or underflow while achieving maximally uniform video quality, as mentioned in [19].

Typical functions of a rate controller are determining a suitable bit rate allocation and its associate Quantization Parameter (QP). This premise has been stated in [12, 19, 20]. However, for the video coding to encode the video data (to compress and at the same time generate certain data rate for the video) is normally achieved by the QP value [21, 22]. The studies on the rate control had utilized available information in order to produce an adaptive video rate which is suitable according to the current network bandwidth variabilities. Some researchers manipulated the available information only at the sender side; these include the generated bit rate and buffer status. Meanwhile, some of the other researchers used a feedback information from the network bandwidth status.

One of the novel algorithms for the rate control, which is called Shaped VBR (SVBR), was created by Hamdi et al. as described in [23]. This algorithm is designed for real-time video application. SVBR is a preventive traffic control which allows VBR coding video traffic directly into the network, while regulating unpredictable large bursty traffic by utilizing a leaky bucket algorithm. Thus, it restricts the excessive bursty media data. SVBR algorithm uses prediction in calculating the next Group of Picture (GoP) video data size and in determining the next appropriate quantization parameter value. This method can be regarded as suitable for real time video application. However, it might produce undesirable results in many video cases, as discussed thoroughly in Chapter four.

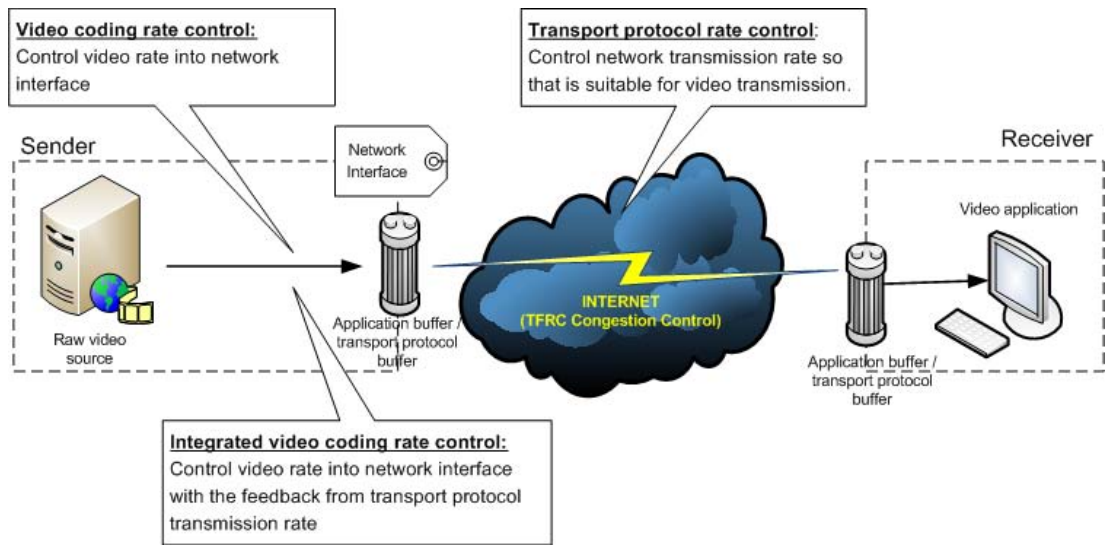
This algorithm has been utilized by many researchers and implemented in many network scenarios. Since SVBR has been introduced, there were several works or improvement, in terms of rate control, that had been done. For instance, work by Lie and Klaue as in [2] have implemented Evalvid-RA, a tool-set for rate adaptive SVBR video performance evaluation in ns-2, based on Evalvid version 1.2 and Evalvid-NS2 [24]. Besides implementing Evalvid-RA, which is based on SVBR principle, they also introduced an integrated rate control algorithm. This integrated rate control gets the feedback from transport layer protocol in order to produce better overall video transmission. They introduced a proprietary Active Queue Management (AQM) feedback from the router, namely P-AQM, to produce a more accurate feedback on the network status. However, this algorithm still works under the limitation of the SVBR.

### **1.1.2 Video Transmission Rate Control Schemes**

There are three schemes of rate control for video transmission; involving controlling the video rate coding, regulating the transport protocol, and the integrated control scheme. Figure 1.4 illustrates the three schemes of the rate control for video transmission.

In the first scheme, namely video coding rate controller, the video data source is regulated so that in the network interface there will be fewer burst. The research works done on in this scheme are presented in [25, 26, 27, 28, 12, 13].

In the second scheme with the transport protocol rate control, network congestion is regulated. This can determine a suitable network speed rate for the video transmission. For all existing transport protocols' congestion control, the principle is that any increment or decrement in transmission rate must be gradual and smooth to ensure the stability, so as to not damage other connections in the Internet. This principle is appropriate for traditional applications, such as http, ftp, and e-mail, that seek to make reliable and effective use of network capacity without requiring short-delay



**Figure 1.4:** Three rate control schemes for the video transmission application

transmission (timely delivery). Thus, it is an obvious contrast between video media transmission rate and transport protocol congestion control to enable better, smoother video streaming over the Internet. Some of the research works done on this scheme are described [29, 30, 31, 32, 33].

For the third scheme, the integrated video coding rate controller gets feedback from the transport layer to produce better performance for the overall transmission as explained in [34, 35, 9, 36].

### 1.1.3 Video Performance Evaluation

The other issue is that most of the studies had relied on network performance metrics only, and they may not produce adequate evaluation to the result of the video application research. There are growing concerns in the video application research to use better video evaluation framework for better perceived quality video measurement, such as the use of Mean Opinion Score (MOS), psycho-visual, or Peak Signal-to-Noise Ratio (PSNR) [2].

## 1.2 Research Problem

As stated previously, there is explosive growth of media traffic traversing the Internet in recent years, but with great challenges as a result of high irregular data produced by media traffic into the Internet. Naturally, video traffic is a variable bit rate data source that generates highly bursty traffic (VBR traffic). Recent implementations mostly buffer the media source in order to regenerate it in the form of constant bit rates (CBR traffic). Consequently, it adds more delays to the system, and thus, it is unable to support the original nature of the video data. Hence, there is a clear need for an alternative solution by taking advantages of both CBR and VBR. By that, the video should be encoded with an open loop VBR as much as possible, but at the same time need to control traffic admission into the network without causing extra delays.

For that, Hamdi et al. as in [23] introduced the Shaped VBR (SVBR). The main idea behind SVBR is to limit the open-loop burst while, at the same time, allowing open-loop VBR coding, provided that they are still within a permitted constraint.

However, since SVBR is created based on estimation or prediction in determining the next GoP of video rate and QP value, it inherits several constraints and weaknesses. Besides its strengths, the SVBR algorithm inherits problems in several circumstances, such as the occurrence of an unwanted sharp increment/decrement in the VBR video rate, the occurrence of a bucket overflow, the existence of a low video rate with low bucket fullness level, and the generation of a cyclical negative fluctuation.

This thesis proposes a new shaped VBR algorithm which is capable of producing a high video rate and at the same time provides a better QP stability video sequence. Besides that, the video rate will be shaped thus there will be no occurrence of unwanted sharp increment or decrement. In addition, the occurrence of the bucket overflow is almost completely reduced.

## **1.3 Research Question**

Given the mentioned research problems of the video transmission issues, a fundamental research question is: how to shape video rate in order to produce video admission rate with no unwanted sharp increments or decrement, thus it can prevent overflow or underflow. Complicating the problem further, the mentioned issues have to be achieved through production of a high quality video, either in terms of high video rate and/or better video quality stability.

## **1.4 Research Motivation**

Video application over the Internet is one of the most interesting applications, where in recent years, an explosive growth through the Internet has been seen. Due to the surge of media traffic over the existing best-effort Internet, network congestion is projected to worsen, as stated in [31]. Video transmission via the Internet is a continuous stream of video data sent over the Internet. However, video media have strict delay and jitter requirements. Thus, it is a very challenging issue here, due to the dynamic uncertain nature of the Internet, especially in terms of variable available bandwidth and random packet losses.

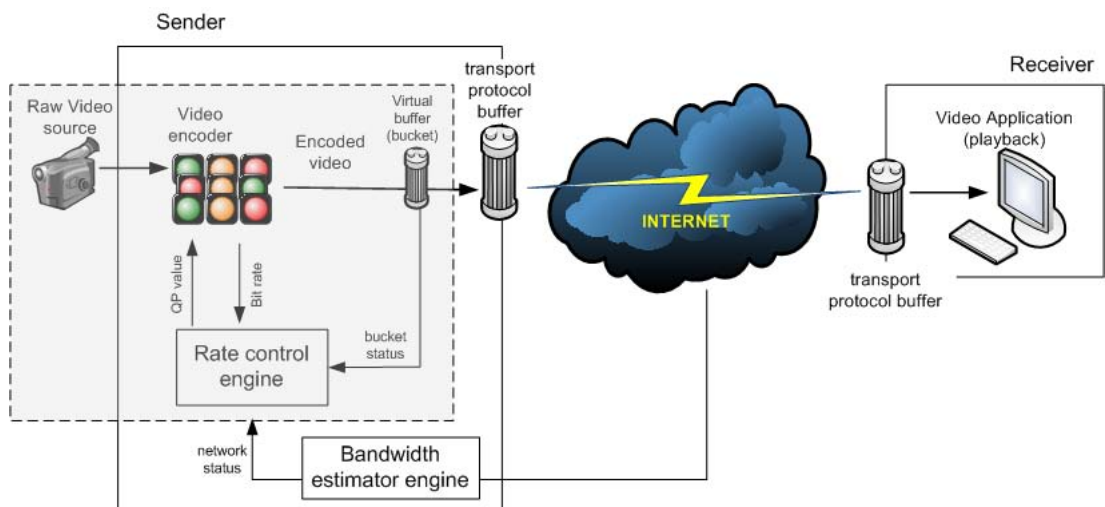
The challenges are greater if one wants to transport high quality real-time video data. High quality video data produces a greater amount of information, thus it demands higher speed, less delay, and better network utilization. High quality video also requires that all video transmission systems or components are working well.

Rate control has reshaped the concept of visual communication and home entertainment, and greatly reduced the traffic load for internet transmission. It has been one of the leading factors in the revolution of information technology, and is continuing to play an important role in the future.

Furthermore, even though many studies and improvements had been introduced and implemented to optimize further the Internet for video transmission application, most of the studies relied on network performance metrics only. It may not produce adequate and realistic evaluation on video application results. This study is also motivated by the lack of studies on the controlled-VBR for slight delay in user-perceived quality video performance.

## 1.5 Scope of Research

As described in Subsection 1.1.2, there are three main work areas in rate control for video transmission. The research presented in this thesis focuses on the first area, which is the video coding rate control. However, since this study employs a leaky bucket algorithm, the rate controller can be integrated with bandwidth estimator engine to produce an integrated rate control system. The light shaded and dotted rectangle area in Figure 1.5 illustrates the research scope. Although the video encoder is included in the scope, it does not mean this research focuses to all aspects of video encoding functions. In this research, the video encoder only acts as a video compressor after receiving a QP value for a GoP of video sequence.



**Figure 1.5:** Scope of the research - video coding rate control only

This study is intended for a slight delay real-time video application. This can be translated into applications which can tolerate a few seconds delay, for instance a few seconds delayed live telecast.

This research is also designed for one-way video application. Thus, two-way application, such as video conferencing is excluded. It also does not consider video VCR function, for instance, fast-forward or backward. However, the proposed solution might be applicable to other applications beyond the scope.

## **1.6 Research Objectives**

The aim of this research is to create a new shaped VBR algorithm for video application based on the Shaped VBR algorithm. The new shaped VBR algorithm is intended to be applied in any video application which can tolerate a slight delay. This new algorithm is designed to eliminate the weaknesses of SVBR and exploit its advantages while enhancing the video application. This research aim has given rise to the following research objectives:

- i. To perform empirical evaluation of the Shaped VBR. This extensive analysis can reveal the strengths and the weaknesses of the algorithm. Consequently, this leads to a set of potential enhancement to the original SVBR algorithm, to be applied in a new shaped VBR algorithm for video transmission application.
- ii. To design a new shaped VBR rate control algorithm for video application based on the Shaped VBR. This new shaped VBR algorithm design is based on the previous finding, which enhances the strengths of the SVBR algorithm and reduces its weaknesses.
- iii. To implement and evaluate the new rate control algorithm performance. Thus, the new algorithm can be tested and evaluated in order to measure its performance.



## **1.7 Key Research Steps**

To accomplish the goal of this research, the following research steps were carried out:

- i. Perform an in-depth study on the video transmission rate control problems and the surrounding issues.
- ii. Perform an in-depth study on the SVBR algorithm to identify the strengths and weaknesses of the algorithm and explore the areas that should be eliminated or enhanced for video application.
- iii. Perform an in-depth study on video performance evaluation, especially on the objective of user-perceived video quality, and its implementation in Network Simulator 2 (ns-2).
- iv. Design a new shaped algorithm for video application. The design processes involve determining the design requirements, objectives, specifications, and justifications for the new algorithm.
- v. Implement, validate, and verify the design of the new shaped algorithm.
- vi. Perform performance evaluation of the new algorithm by comparing it with the SVBR algorithm.

## **1.8 Key Contributions**

The important contributions of this research are presented in the following points;

- The empirical evaluation of the SVBR algorithm, which identifying the strengths and weaknesses of the algorithm.

- A new shaped VBR algorithm for video application is proposed based on the analysis of SVBR. Specifically, this novel algorithm, as compared to SVBR, leads to the following effective contributions:
  - Effectively minimize the estimation or prediction of the video rate that leads to unwarranted result in the SVBR algorithm.
  - Improve the overall video rate that leads to a better video quality. The improvement of the video rate is effectively shaped without the risk of data loss as a result of buffer overflow.
  - Enhance the video visual stability by stabilizing the Quantization Parameter (QP) value.
- The implementation of the new mechanism of the rate control for video application in a network simulator (ns-2) and it is also capable of working with different Group of Picture (GoP) video sizes.
- The development of the algorithm in an Evalvid environment, which is capable of performing user-perceived video performance evaluation.

## 1.9 Organization of the Thesis

This thesis is organized into seven chapters as follows:

**Chapter 1** provides a broad overview of the thesis. It presents a compact introduction to the issues surrounding video transmission and rate control. The other significant aspects which have been written in this chapter are the research properties, such as the objectives, scopes, contributions, etc.

**Chapter 2** specifies a technical background of the research work. It lays down some background information that is relevant to this research and sets the context of the research work. Then, related research work is discussed analytically.

**Chapter 3** discusses the methodology and simulation setting employed in this research work. The deliberations are on the methodology options, the challenges to fully grasp the behavior and performance issues of a protocol without evaluating it under controlled conditions, and the chosen method of performing the research performance evaluation. In addition, the simulation settings are elaborated extensively in this chapter, which includes specifying the justifications and how all setups can closely match the real Internet environment.

**Chapter 4** deliberates on the extensive analysis and empirical evaluation which are conducted on the SVBR algorithm. It describes comprehensively the SVBR algorithm, in terms of its principles, on how it determines the suitable bit rate and QP value. Also, this chapter presents the experiments conducted to compare the SVBR performance with traditional rate controllers, namely Variable Bit Rate (VBR) and Constant Bit Rate (CBR). Finally, the SVBR algorithm is empirically analyzed to identify its strengths and weaknesses. From the analysis, the important areas which can be enhanced are highlighted.

**Chapter 5** elaborates on the design of a new algorithm for video application. The elaboration is on design justifications, specifications, requirements, and objectives for the new algorithm. Then, the detailed design is discussed in depth. Also, this chapter covers the implementation of the new algorithm, which includes the work environment, the generation of the video trace files, the simulation package, the coding, and the verification and validation.

**Chapter 6** thoroughly discusses the performance evaluation of the new algorithm as compared to the SVBR. It shows the extensive analysis of the behavior of this algorithm in several specific conditions.

**Chapter 7** states and discusses the global conclusions of the research work presented in this thesis and provide some suggestions for further studies.

# **CHAPTER TWO**

## **BACKGROUND AND RELATED WORK**

This chapter provides background information and sets the context of the work presented in this thesis. The literature that relates to this research is covered thoroughly and analytically. Following this section, some background information relevant to this research is described. It is rather compact but comprehensive since the research focus is characteristically somewhat multi-domain.

The issues to be discussed in this chapter are the challenges in transporting video over the Internet, video coding, video rate control, network transmission protocol, works related to the video rate shaping algorithm, and etc. In the challenges in transporting video over the Internet section, it will be highlighted the phenomenon of explosive growth of media traffic traversing the Internet, however the Internet is still inherent with the problems in accommodating these kinds of application. In the video coding section, the explanations are on how the video data is calculated and its hierarchy. The section also will cover the issues, standards, and techniques of video coding.

In the video rate control section, the elaborations are on the role of video rate control in the video transmission system, the relationship of video rate control and video coding, the relationship of QP and the rate control, the traditional rate controls,

the Rate-Distortion (R-D) theory, and researches in video rate control. In network transmission protocol section, the discussions are on the transmission protocols to carry the video data. This protocol includes the two popular protocols in the Internet, which are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). The discussion is on the suitability of the protocol in transporting video, and an alternative protocol is discussed as well. Finally, in the work related to the video rate shaping algorithm section, the related research and progress in video rate shaping algorithms will be analytically discussed. This includes the early and recent works in this research domain.

## **2.1 The Challenges in Transporting Video over the Internet**

Surging in growth of media traffic crossing the Internet has been witnessed in recent years. Various real-time applications are becoming increasingly common and must be accommodated. The deluge of video media applications over the Internet is attributed to the increasing capacity of the Internet and its cost-effectiveness. However, video tends to congest the computer network, and the network congestion condition is projected to worsen. Thus, the networking research community has witnessed a high uprising in research on all aspects of video transmission [37].

Bauer et al. [38] stated that the growth in video traffic is definitely significant, and eventually will fundamentally reshape the increasing availability of various traffic on broadband networks. Thus, according to them this will not only affect the high network bandwidth utilization, but it raises various new and difficult challenges, for instance, in terms of predicting per user usage patterns over time and for categorizing users into usage-types. This also increases uncertainty and makes it more difficult to manage congestion relying solely on either the per-flow TCP mechanisms or the longer-term capacity expansion. Thus, it is not surprising that network operators

confronted with this more complex environment have opted to explore a range of new traffic management techniques.

To reduce the bandwidth of video for transporting over the Internet, the video is typically encoded or compressed. However, as stated by many authors [39, 40, 41, 42], that even with a compressed video, it still requires a large bandwidth of the order of hundred kbps or Mbps. Another challenge is compounded by the fact that video sequences comprise widely varying content and motion, as stated in [43]. In addition, as mentioned in [44, 45], a compressed video usually demonstrates a highly variable bit rate property, and with bandwidth uncertainties such as loss and delay, makes the transport of video over the Internet a great challenging issue. This includes the characteristics of video, video traffic modeling, video coding, and video rate shaping as well as protocols and mechanisms for the efficient transport of video, all of which have received a great deal of interest among networking researchers and network operators where diverse video transport schemes have been developed.

## **2.2 Video and Coding**

A video scene is a series of still images that change so fast that it looks like the image is moving [46]. Thus, a video sequence is displayed at a certain frame rate. Commonly display rates are 30 and 25 frame per seconds (fps), particularly in normal TV display. The National Television Standards Committee (NTSC) format uses 29.97fps, or approximately 30fps. Whereas, the Phase Alternating Line (PAL) employs 25fps [47].

However, to transmit a raw video over the Internet, its size is very large. As discussed previously, many efforts need to be done. While the whole video transmission activities are rather complicated (refers to 1.1 and 2.3.1), but among the essential work that needs to be done is to compress (encode) and to control the video transmission rate. The following subsections are the description on video and

its coding, in the purpose of facilitating and setting the context of video rate control and/or rate shaping related studies discussion.

### **2.2.1 Video Sequence**

As mentioned previously, a video scene is a series of still images, which create a motion display. Each image or frame consists of picture elements, technically known as pixels or pels. The video frame format specifies the size of the individual frames in terms of pixels. For instance, in typical TV format, such as in ITU-R/CCIR-601 format, has  $720 \times 480$  pixels. It means that the frame is constructed of a 720-pixel width and 480-pixel height. The usual frame size for a typical frame format for video transmission is  $352 \times 288$  pixels for Common Intermediate Format (CIF) format, and  $176 \times 144$  pixels for Quarter CIF (QCIF) format [48].

Briefly, each pixel needs to be represented by several bits of data. Two most popular representations for the pixel are chrominance (chroma or C for short) and luma (Y for short) to represents the brightness in an image. Chrominance is the signal used in video systems to convey the color information of the picture. It is represented by two color components, identified as the U-V color plane, and is defined by the three chromaticities of the red, green, and blue, which are popularly known as RGB. While, Luma represents the achromatic image without any color, the chroma components represent color information. Each pixel is typically a combination of chrominance and luma, which commonly known as YUV [49].

The more bits used in representing a pixel means the higher video frame quality. However, the more bits used also means that high resources are needed, either in terms of storage space or network bandwidth. A normal TV system uses 24 bits per pixel, and CIF/QCIF is typically represented by 8 bits. Uncompressed video produces an enormous size of data. For instance, in a simple calculation, one second of NTSC video display is equal to 26MByte size of video ( $640 \times 480$  [pixels per frame]  $\times 3$  [bytes

per pixel] x 30 [frames per second]). This size grows to 1.6 GByte for one minute and 96 GByte for one hour.

For Internet transmission, the video rate calculation is usually done in bits per second (bps). For a small screen, a  $352 \times 288$  CIF video frame with 8 bits per pixel, and a frame rate of 25fps, the video rate is 20.3Mbps. This obviously cannot be supported by typical home broadband speed nowadays, which is only 1 Mbps. With this huge size of data, or video rate, it is impossible with the present Internet bandwidth to support the video transmission. Hence, the video coding (compression) is necessary before the transmission.

### **2.2.2 Video Hierarchy**

Video is organized in a hierarchical structure; Video Sequence, Group of Pictures (GoP), Frames, Slices, Macroblock, and Block [44]. There are other categorizations according to data granularity of video hierarchy. However, since they are not common, they are not going to be discussed here. By knowing this hierarchy of video, the complexity of a video rate control research, and the delay of overhead processing can be gauged. The lower the video granularity in the hierarchy, the higher it is in complexity and overhead processing.

#### **2.2.2.1 Video Sequence and GoP**

A video sequence can be regarded as a video clip or a movie. It begins with a sequence header, then followed by many video frames, possibly hundreds or thousands of frames. This video terminates with an end-of-sequence code. The video frames are usually grouped into GoP, where the number of frames for each GoP can be 9, 12, 15, or more.














#### **2.2.2.2 Frame, Frame Type, and the Relationship with GoP**

The frame is a primary coding unit of a video sequence. As mentioned previously, a frame size is measured in pixels, and each pixel is represented by a certain number of bits. Besides that, a frame is identified by its type. There are three typical types of video frames, which are I-frame, P-frame, and B-frame, and normally, each GoP can consist of I-, P-, and B- frames [50]. These frame types play an important role in video coding or compression, which will be explained in Subsection 2.2.3.

As explained in [50], I-frame or Intra-frame is a reference frame or the frame that is coded without prediction from other frames, or it is coded using only information present in the frame itself. On the other hand, the P-frame or Predicted-frame is a frame which is coded with respect to the nearest previous I- or P- frame. This frame can serve as a prediction reference for B-frame and future P-frame. Finally, the B-frame or Bidirectional frame is a frame that uses both before and afterward frame, as its reference. The frame which less refers to another frame, particularly the I-frame, is the least which can be compressed. On the other hand, the frame which refers most to another frame, is the most potential frame which can be compressed with a high degree of compression. However, the frame which refers less to others, is the most important frame.

In a GoP, an I-frame is a must-be-present frame, P-frame is usually available, and B-frame might not exist. A GoP starts with I-frame and ends with P- or B- frame. For a GoP with nine frames, the arrangement may look like this: IBBBPBBBBP. However, it does not always sets in that kind of arrangement. For instance, Figure 2.1 shows the GoP arrangement which has been used in this study, which is a GoP with 12 frames consisting of I- and P- frames only.

Frame #	1	2	3	4	5	6	7	8	9	10	11	12
Video frame												
Frame type	I	P	P	P	P	P	P	P	P	P	P	P

**Figure 2.1:** The GoP arrangement of frames type used in this research

### 2.2.2.3 Block, Macroblock, and Slice

A block is the smallest coding unit in the video hierarchy, which consists of 8x8 pixels. A macroblock is a 16x16 pixel or a group of four blocks in a square of 2x2 blocks. Meanwhile, a slice contains one or more contiguous macroblocks. The order of the macroblocks within a slice is from left to right and top to bottom. A single frame is divided into slices, which represent independent coding units that can be decoded without referencing other slices of the same frame.

### 2.2.3 Video Coding

Video coding or video compression refers to the process of reducing the quantity of video [51]. The quantity reduction can go up to ten times of the original quantity. Compression is made in the expense of quality. Lower quality, i.e. smaller resolution, frame rate, and imprecise representation of image pixels, are, however, enough for many purposes. Thus, video coding plays an important role in bridging the gap between large amounts of visual data and limited bandwidth networks for video distribution [52].

A number of standards of video coding have been defined, such as MPEG-2, H.263, and MPEG-4. Video coding is performed by hardware or software solutions, generically called coder/decoder or codec in short. Codec can effectively reduce the bandwidth required to transmit a digital video. For example, a codec standard, called MPEG-2, can compress two hours of video by 15 to 30 times while still producing a picture quality that is generally considered a high quality for standard-definition video.

There are many video coding standards for different purposes of video application and different complexities/flexibilities/advancement for compression capabilities. Table 2.1 summarizes some of the prominent video codings in the industries.

**Table 2.1:** Video coding standards

Name	Completion Time	Major Features
H.261	1990	For videoconferencing, 64Kbps-1.92 Mbps
MPEG-1	1991	For CD-ROM; $\leq 1.5$ Mbps.
MPEG-2 (H.262)	1994	For DTV/DVD, 2-15 Mbps; for ATSC HDTV, 19.2 Mbps; most extensively used standard.
H.263	1995	For very low bit rate coding, below 64Kbps.
MPEG-4 Part 2	1999	For multimedia, content-based coding, its simple profile and advanced simple profile are applied to mobile video and streaming
H.264/AVC (MPEG-4 Part 10)	2005	For many applications with significantly improved coding performance over MPEG-2 and MPEG-4 part 2

### 2.2.3.1 Inter- and Intra-Frame Coding

There are various ways of performing video coding or to compress the video. The main idea of video compression is to remove the redundancy in a video [44]. The redundancy can be looked at either the redundancy in (within) a frame or redundancy between video frames [42]. Therefore, the video coding ways can be categorized into two groups, which are;

- Intra-frame or within a frame, which is also known as spatial encoding. Spatial or Intra-frame encoding is performed by taking advantage of the fact that the human eye is less able to distinguish small differences in color than it can

perceive changes in brightness. Therefore, the almost similar areas of color can be regarded as one color or be represented by a minimum number of colors.

- Inter-frame of between frames, which is also known as temporal encoding. In temporal compression or Inter-frame coding, only the changes from one frame to the next are encoded.

### **2.2.3.2 Video Coding Techniques**

Typically, intra-coding yields a smaller compression ratio compared to inter-frame coding. This is because the most important type of redundancy comes from the repetition of almost identical successive images. The most efficient compression is, therefore achieved by utilizing the correlations or similarities between successive images [53]. However, as mentioned previously, intra-coding is more commonly used because of its low complexity. Nevertheless, the detailed technique of the video codings is complex. Thus, the following subsections explain the techniques briefly.

#### **Intra-frame Coding Technique**

In intra-frame coding, each video frame is divided into blocks of  $8 \times 8$  samples. Each block is transformed using the DCT into a block of  $8 \times 8$  transform coefficients, which represents the spatial frequency components in the original block. These transform coefficients are then quantized by an  $8 \times 8$  quantization matrix that contains the quantization step size for each coefficient. The quantization step sizes in the quantization matrix are obtained by multiplying a base matrix by a quantization scale. This quantization scale is typically used to control video encoding. A larger quantization scale gives a coarser quantization, resulting in a smaller size (in bits) of the encoded video frame. The quantized coefficients are then zigzag scanned, run-level coded, and variable-length coded to achieve further compression.

## **Inter-frame Coding Technique**

As discussed previously, there are various ways a codec compresses video. In inter-frame compression, a typical codec operates on a macroblock. These macroblocks are compared from one frame to the next, and the codec will send only the differences within those macroblocks. In a fast scene, more macroblocks differ from the previous macroblock frame, so the codec must produce more data. In other words, in a fast-moving scene, the compression technique is less effective. From the other perspective, if a video consists of many interchangeably fast and slow scenes, there will be frequent bursty traffic produced. The parts of the macroblocks that are not changing need not be sent repeatedly. Consequently, it compresses the video more effectively than others.

The macroblock comparison process is called a block matching algorithm. If the encoder succeeds on its search, the block could be directly encoded by a vector, known as a motion vector, which points to the position of the matching block at the reference frame [54]. The process of motion vector determination is called Motion Estimation. Although in most cases the encoder will succeed, the results will not be accurate because the block found by the encoder will be similar to but hardly exactly the same as the block it is encoding.

To sum up, if the encoder succeeds in finding a matching block on a reference frame, it obtains a motion vector pointing to the matched block and prediction error. Using both elements, the decoder is able to recover the raw pixels of the block.

### **2.2.3.3 Advancement in Encoding Techniques and Standards**

There are many emerging of video encoding techniques and standards to provide flexibility and capability in catering for various recent video applications. The minus side of this development is on the encoding complexity, which may contribute an additional processing delay.

## **Scalable Video Coding (SVC)**

Scalable Video Coding (SVC) [55, 56] refers to the encoder that typically produces multiple layers of a video frame data; mainly, the base layer and the enhancement layer. The base layer provides a basic quality, such as in terms of low spatial or temporal resolution video, while the enhancement layer improves the video quality, for instance, by increasing spatial resolution or frame rate. One of the popular standards is H.264/MPEG-4 Part 10 or Advanced Video Coding (AVC). The standard was developed jointly by Telecommunication Standardization Sector of the International Telecommunications Union (ITU-T) and International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC), ISO/IEC JTC 1. These two groups created the Joint Video Team (JVT).

Scalable encoding is a convenient way to adapt to the wide variety of video-capable hardware, for example, PDAs, cell phones, laptops, and desktops, and the delivery networks, for instance, wired versus wireless. Each of these devices has different constraints due to processing power, viewing size, and so on. Scalable encoding can satisfy these different constraints with one encoding of the video.

SVC standardizes the encoding of a high-quality video, which contains one or more subset bitstreams. A subset video bitstream is derived by dropping packets from the larger video to reduce the bandwidth required for the subset bitstream. The subset bitstream can represent a lower spatial resolution, lower temporal resolution, or lower quality video signal. Spatial resolution refers to number of pixels in the video screen or the screen size of the video frame, while temporal resolution refers to the video frame rate.

An alternative to scalable encoding is to encode the video into different versions, each with a different quality level, bit rate, and spatial/temporal resolution. The advantage of having different versions is that it does not require the more sophisticated scalable encoders and does not incur the extra overhead due to the scalable encoding.

The drawback is that the multiple versions take up more space on servers and possibly need to be streamed altogether over the network to be able to choose the appropriate version at any given time.

Transcoding is another alternative to scalable encoding. Transcoding can be used to adapt to the different network conditions, or to adapt to different desired video formats. The transcoding approach requires a typically high performance intermediate node [57].

### **Fine Granularity Scalability (FGS)**

Fine Granularity Scalability (FGS) [58] is a relatively new form of scalable video encoding that has recently been added to the MPEG-4 video coding standard. The main purpose of the FGS is to enhance the flexibility of a scalable encoding, particularly for video streaming. With FGS, the video is encoded into a base layer (BL) and several enhancement layers (ELs). The server can choose the number of ELs to stream to the client in addition to the BL, as a function of network conditions. However, with FGS coding, any number of bits of the FGS-encoded EL can be suppressed at the server before transmission, and the achieved image quality is directly proportional to the number of bits decoded at the client.

Similar to conventional scalable encoding, the FGS enhancement layer is hierarchical in that “higher” bits require the “lower” bits for successful decoding [59]. This means that when cutting the enhancement layer bit stream before transmission, the lower part of the bit stream (below the cut) needs to be transmitted, and the higher part (above the cut) can be dropped. The FGS enhancement layer can be cut at the granularity of bits. The flexibility of FGS makes it attractive for video streaming, as video servers can adapt the streamed video to the available bandwidth in real-time, without a heavy re-encoding process. However, this flexibility comes at the expense of reduced coding efficiency.

## **Multiple Description Coding (MDC)**

In Multiple Description Coding (MDC) [60], video is encoded into several sub-streams referred to as descriptions. Each of the descriptions is conveyed toward the receiver. Decoding more descriptions gives a higher video quality while decoding an arbitrary subset of the descriptions, resulting in lower quality. The packets of each description are routed over multiple, (partially) disjointed paths. The individual descriptions have no explicit hierarchy or dependency among them, i.e. any combination of the descriptions can be combined and decoded. This contrasts with conventional hierarchical layered videos where a received enhancement layer is useless if the corresponding base layer is missing, as it is for FGS.

Besides increased fault tolerance, MDC allows for rate-adaptive streaming. Briefly, it works as follows; content providers send all descriptions of a stream without paying attention to the download limitations of clients, and receivers who can not sustain the video rate, only subscribe to a subset of these streams, thus freeing the content provider from sending additional streams at lower video rates.

MDC can be seen as another way of enhancing error resilience without using complex channel coding schemes. The goal of MDC is to create several independent descriptions that can contribute to one or more characteristics of video: spatial or temporal resolution, signal-to-noise ratio, and frequency content.

## **Wavelet-Based Video Encoding**

With wavelet transform coding, a video frame is not divided into blocks as with the DCT-based Moving Picture Experts Group (MPEG) coding. Instead, the entire frame is coded into several subbands using the wavelet transform. The wavelet transform has many advantages over the DCT transform. The most obvious of them all is the compact-support feature. This compact-support allows the translation of a time-domain function into a representation that is not only localized in frequency, but



in time as well. The net result of this is that the wavelet transform can occur over the entire image within a reasonable computational period and bit DCT-based transforms are eliminated in the wavelet transform.

## 2.3 Video Rate Control

According to Tian [61], video rate control refers to coding a video so that the resulting bitstream satisfies a target bit rate. Rate control plays an important role in video coding and transmission [12, 62, 14]. This is true because the typical transmission network is not able to provide Quality of Service (QoS) and its transmission rate varies due to the shared nature of IP networks [9]. Hence, video transmission rate needs to be regulated to adjust it according to the network condition or constraint. Therefore, rate control is essential in video transmission for an accepted visual quality under some certain rate constraint. This is due to the fact that overly coding a high video rate can cause an undesirable traffic burstiness and exceed the capacity of channel. While on the other hand, uncontrolled reduction of the output bit rate of a video coder leads to unnecessary quality degradation and inefficient use of available bandwidth resources.

According to Chen and Ngan as in [14], in formulating a good rate control algorithm, at least the three following challenging issues need to be considered:

- **Distortion:** There is an inherent tradeoff between distortion and bit rate. Based on the Rate-Distortion (R-D) theory, which will be described later, the distortion  $D$  is a decreasing function of the bit-rate  $R$ . A decreasing distortion leads to an increased rate and vice versa. In other words, rate control should balance the maximum picture quality (minimum distortion) without exceeding the maximum permitted bit rate.
- **Complexity:** Different applications have different computational complexity requirements. For a real-time video application, a rate control algorithm

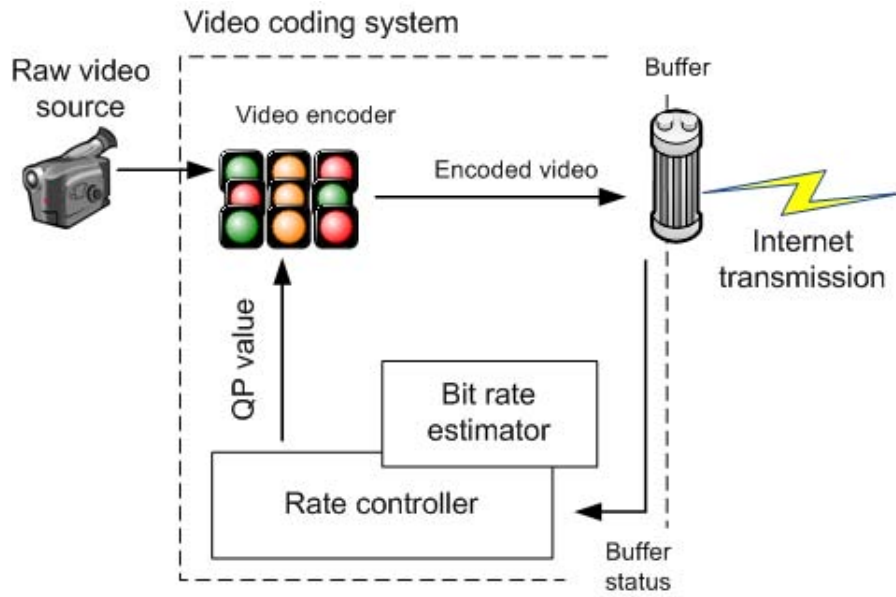
should reduce the delay to a very minimum level by applying a simple rate control algorithm. Two-pass rate control algorithm, which is doubled overhead processing, is not applicable to real-time applications, but it may be used for delay-based application, such as Video on Demand (VoD).

- **Constraints:** There are various constraints for video compression applications. For end-to-end real-time video communication systems, there is the delay constraint to avoid delay jitter and jerky motion. For constant bit-rate applications, buffer constraint is introduced, where the rate control algorithm should guarantee that the encoder and decoder buffers do not overflow or underflow.

### 2.3.1 Rate Control and Video Coding

Video coding and rate control are closely related. Most video coding standards do not specify the rate control algorithm as its native part; it is generally an informative part in video coding standard [62, 14, 63]. Moreover, most encoders have no strategy for reacting to bitrate variations that occur as a result of changes in network condition [9]. Therefore, the implementation of a rate control algorithm is left open to the designers to develop the algorithm based on the video application requirements. However, as mentioned previously, rate control plays an essential role in video coding, especially for transmitting video over the Internet. As a key component in video coding, rate control has been studied extensively. Figure 2.2 illustrates the relationship of a rate controller and a video coding.

The objective in video coding and transmission is to provide the best video quality for the users, giving the constraint of certain network conditions. This means that if the network condition is congested or has limited bandwidth, the video rate (the video quality) has to be reduced. However, the reduction should still deliver the optimum quality which the network permit.



**Figure 2.2:** Relationship between rate controller and video coding

The rate controller works as follows; when a raw video is going to be transmitted, a rate controller calculates or estimates a suitable targeted bit rate for the data. Then it generates a suitable QP value for the video encoding engine to compress the raw video. A raw video might be in GoP, macroblock, or in other video granularities, depending on the video coding system or video rate controller. The video encoding engine uses a QP value, compresses the raw video, and then produces a compressed/encoded video which is usually much smaller in size than the raw data. This smaller size video also means that it produces a slower video rate. Whether the generated video rate is same as the targeted bit rate of the rate controller, it is another challenging issue in the video rate control research. Another challenging issue here is that these rate adaptation methods not only lead to variable video quality, but occasionally may cause poor utilization of bandwidth because of selecting too high a QP value, which produces too low video rate.

As stated previously, the main function of the rate controller is to optimize the image quality. Optimizing the image quality means minimizing the video coding artifact (or distortion) of the reconstructed video at the receiver end. Image quality

refers to the number of bits in presenting the image or the spatial detail of a video frame. The coding also means a reduction of “video quality” by removing certain “redundant” bits in a video frame. The more redundant bits involved means the better the compression will be, and it also contributes to a reduction of the video quality.

For the frames that contain lots of details and motion, video rate goes up quickly. Meanwhile for the frames that do not contain many details and motion, the rate will be very low. As a result, the overall bit rates are not predictable, which is not desirable in practice. Therefore, based on feedback from a network parameter (particularly in terms of current network transmission rate) or buffer/storage state, the rate controller engine calculates a suitable QP value for the next unit of video transmission.

### **2.3.2 Understanding the Relationship between Quantization Parameter (QP) and the Rate Control**

MPEG-2 [64] video encoding offers a rich array of possible methods for video rate control. A fundamental parameter is the Quantization Parameter (QP) value or scale, since it controls the instantaneous bit rate used at the macroblock level. The QP scale is used to control the resolution of the DCT coefficients. The QP scale can be set globally at the frame layer, and the desired rate can be achieved by varying the scale for different types of pictures. The QP value can also be adjusted at the slice or macroblock layer to provide finer granularity in the rate control.

As discussed previously, one of the typical ways in regulating the video bit rate for the transmission is by adjusting the QP scale. The higher the QP scale means the lower the video quality. Table 2.2 shows a sample of QP values and their corresponding video rate (in Bytes) for several GoPs. This sample has been generated from a video sequence used in the experiment by using Evalvid-RA tool.

For example, in GoP 1 when  $QP = 2$ , the video rate is 78626 Bytes/Gop, and when  $QP = 3$  the video rate is 52416 Bytes. The higher the video rate means the higher the video quality. When the network is estimated to be unable to support a high video

rate, the video controller will generate a high QP value in order to reduce the video rate (reduce the video quality as well). Therefore, for instance, when estimating that a network transmission rate is degraded to around 22000 Bytes, the video rate controller should increase the QP value to 8, which will produce a 22176 Bytes/GoP video rate. However, another challenge for video rate research is that the corresponding QP-video rate is different among different video sequences.

**Table 2.2:** QP values and GoP video rate

		Quantization Parameter (QP)								
		2	3	4	5	6	7	8	9	10
Group of Picture (GoP)	1	78624	52416	40320	35280	33264	27216	22176	21168	20160
	2	75600	50400	36288	34272	31248	23184	22176	21168	20160
	3	77616	51408	41328	35280	32256	25200	22176	21168	20160
	4	72576	48384	37296	34272	31248	23184	22176	21168	20160
	5	70560	48384	37296	34272	28224	23184	22176	21168	20160
	6	78624	51408	41328	35280	32256	27216	23184	21168	20160
	7	89712	58464	46368	40320	33264	30240	27216	25200	21168
	8	103824	75600	59472	47376	40320	34272	33264	30240	29232
	9	88704	60480	48384	38304	33264	32256	30240	25200	20160
	10	76608	52416	45360	34272	33264	31248	24192	21168	20160
	11	76608	53424	42336	35280	31248	27216	25200	23184	20160
	12	85680	58464	46368	34272	32256	31248	30240	22176	20160
	13	84672	58464	49392	37296	35280	32256	26208	23184	22176
	14	75600	50400	38304	35280	32256	24192	22176	21168	20160
	15	88704	59472	47376	37296	32256	31248	26208	22176	20160
	16	90720	62496	48384	40320	33264	31248	29232	24192	20160
	17	73584	48384	38304	34272	31248	24192	22176	21168	20160
	18	76608	51408	38304	35280	32256	23184	22176	21168	20160
	19	81648	54432	43344	34272	32256	30240	24192	21168	20160
	20	88704	59472	47376	35280	33264	31248	28224	21168	20160

To the human eyes, it is more suitable to view a video with the same QP values for all video frames because it yields a constant video quality. However, in the aspect of network transmission rate, it is unable to provide a consistent bandwidth to the video application. As a consequence, the video playback might be intermittently halted. In this case, the users prefer to view a continuous playback, even though it is with dynamic QP values (different video quality).

The key issue in rate control is to estimate bits (or bytes, depending on the appropriate data unit) which are suitable for the current status of the network.

This, in rate control research is usually mentioned as the “Rate-Distortion” (R-D) behavior of the video coding. Nevertheless, the R-D behavior is characterized by its “rate-quantization” and “distortion-quantization” functions. In other words, the key point is to find the relation between the video rate and QP, and it is usually developed based on a rate-distortion (R-D) model and R-D theory [65].

### **2.3.3 Rate-Distortion Theory**

According to Chen and Ngan [14], R-D theory is the theoretical foundation of rate control, and it is said to originate from Shannon’s work [66, 67]. The central paradigm of Shannon’s classical information theory is the engineering problem of the transmission of information over a noisy channel. The most fundamental results of this theory are Shannon’s source coding theorem, which establishes that, on average, the number of bits needed to represent the result of an uncertain event is given by its entropy; and Shannon’s noisy-channel coding theorem, which states that reliable communication is possible over noisy channels provided that the rate of communication is below a certain threshold, called the channel capacity. The channel capacity can be approached in practice by using appropriate encoding and decoding systems.

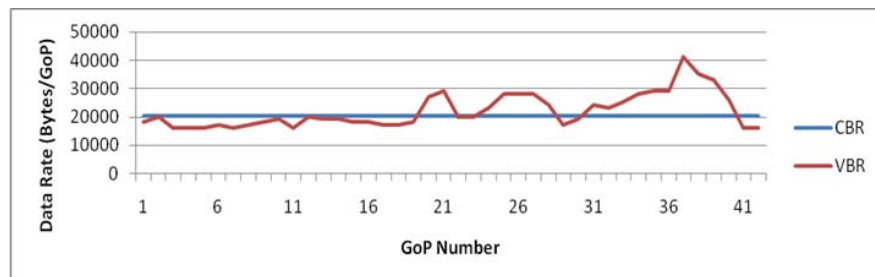
Information theory [68] is closely associated with a collection of pure and applied disciplines that have been investigated and reduced to engineering practice under a variety of rubrics throughout the world over the past half century or more; such as adaptive systems, anticipatory systems, artificial intelligence, complex systems, complexity science, cybernetics, informatics, machine learning, along with system sciences of many descriptions. Information theory is a broad and deep mathematical theory, with equally broad and deep applications, amongst which is the vital field of coding theory. Coding theory is one of the most important and direct applications of information theory. It can be subdivided into a source coding theory and channel

coding theory. Using a statistical description for data, information theory quantifies the number of bits needed to describe the data, which is the information entropy of the source.

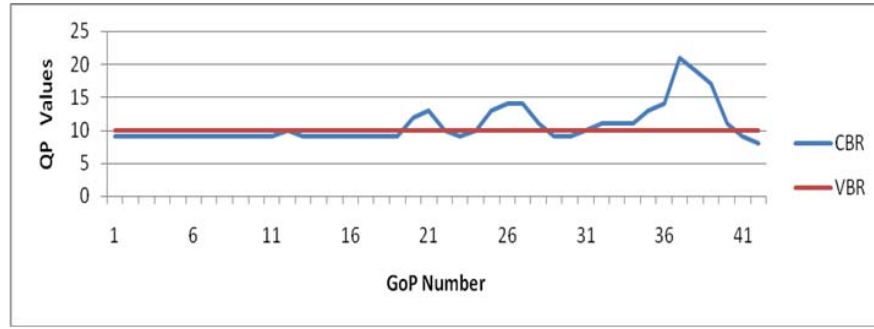
Thus, R-D theory forms a central part of information theory and lossy source coding, which directly relate to lossy image compression and video compression. A lossy source coding scheme, such as video coding, concentrates on the tradeoff between the distortion and bit rate. The basic scenario is that a decreasing distortion (D) leads to an increasing rate (R), and an increasing distortion is achieved by a decreasing rate. In R-D theory, the R-D function is defined to describe the lower bound for the rate at a given distortion.

### 2.3.4 Traditional Rate Controllers: VBR versus CBR

Currently, most of the video applications employ video rate controllers in the form of either a Variable Bit Rate (VBR) or Constant Bit Rate (CBR) [69]. VBR uses the same QP value for all the video microblocks, frames or GoPs, depending on the referred video granularity (in this research, the GoP is used as the video granularity reference). This means that the video rate may be different for each GoP. In the CBR case, the video rate will be constant for all GoPs. Consequently, the QP values might be different for each GoP. The relationship between CBR and VBR, in terms of video rate and QP values, are shown in Figures 2.3 and 2.4.

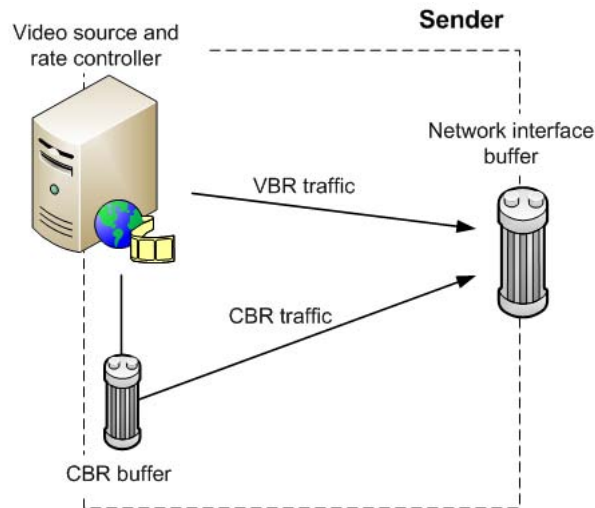


**Figure 2.3:** video rate comparison for CBR and VBR



**Figure 2.4:** Quantization Parameter comparison for CBR and VBR

Digital video is bursty in nature, and the burstiness depends on the frequency of changes, either in the background or the movement of objects in the foreground [70]. Without rate control, the output bitstream of a video encoder will be VBR, since it depends on the complexity of the scene, degree of motion, and frequency of scene changes. Digital video encoder manufacturers accordingly implement additional rate control in the encoder to generate a CBR stream for transmission and distribution applications, usually with the deployment of additional buffer. Figure 2.5 illustrates a typical setup of the VBR and CBR rate controllers.



**Figure 2.5:** VBR and CBR setup

Therefore, both CBR and VBR have their respective advantages and disadvantages. As stated in [71], the advantage of VBR is that it uses the same QP for all GoPs,



which means, it produces a consistent visual quality. On the other hand, the advantage of CBR is that it generates constant video rate for the network interface. However, the problem of VBR is its bursty character, which causes problems to networks, and it leads to considerable variation of the network traffic, jitters and delays. On the other hand, the problem with CBR is its visual quality that tends to vary according to the video content. Besides that, normally CBR involves additional buffer, which causes buffering delay or overall delay. Thus, it can be said that VBR is good for human viewing experience, but it is not “friendly” to the transmission over a network. Conversely, CBR is friendly to the transmission network, but it is not satisfying, in terms of human viewing perception.

### **2.3.5 Rate Control Research**

Most rate control research investigates the constraint of available/allowable bandwidth of the network, and on how to achieve best picture quality or minimum video rate reduction (coding distortion), normally by optimum bit allocation and accurate rate control. In the other words, the bit allocation scheme is employed in order to allocate bits of video among the video frames in such a way that the overall distortion is minimized. The rate control algorithm is then employed to meet the bit’s target by selecting appropriate quantization settings for the video encoder.

According to Jiang [19], the rate control problem has been studied in two sub problems, which are bit allocation and QP determination. In the bit allocation, the target bits are allocated among different coding units such as GoPs, frames, or macroblocks. GoP bit allocation involves selecting the number of bits to be allocated to a GoP. Frame bit allocation involves distributing the GoP budget among the frames, so as to achieve a uniform video quality. Macroblock bit allocation involves tuning the parameter for each macroblock of a frame so that the rate regulations are met, and a uniform quality is achieved within the frame.

In terms of QP determination, QP is calculated to achieve the allocated bit budget. This parameter can be selected for the entire frame (frame layer rate control, in which all MBs within a frame use the same QP for encoding) or changes from macroblock to macroblock (macroblock layer rate control). Both sub-problems have been extensively studied under the framework of rate-distortion theory.

However, Chen and Ngan [14] mentioned that there are three categories of widely used approaches according to the way for calculating the target bit allocation and/or the QP. The first category is buffer-based approaches, which compute the target bit allocation and/or the QP on a macroblock (MB) or frame granularity level, based on the buffer state, the previous bit count, or both. The second category includes the analytic modeling approaches, which perform a calculation on a set of “rate-quantization” and “distortion-quantization” functions, which are derived from the overall statistical properties of the source data. The third category uses the operational rate-distortion (R-D) modeling approach where the assignment is preformed only after the statistics of the signal in future frames or the model parameters estimated from the past data are processed.

According to Chen and Ngan [14], the research in the field of video rate control can be categorized into; (i) R-D modelling, (ii) quality constrained rate control, (iii) from objective optimization to subjective optimization, and (iv) hybrid structure-based optimization. The following subsections are the description of this categorization.

#### **2.3.5.1 R-D Modelling**

This rate control research category is based on R-D theory, where the idea is to find an optimum or sub-optimum of R-D relationship. One of the research, which falls into this category was performed by Cheng and Hang [72] to achieve a more uniform perceptual picture quality. They suggested two important steps, firstly a QP planning strategy or bit allocation scheme, which pre-analyzes the entire image content and then allocate

bits accordingly, and secondly by taking image contents into consideration. However, these two steps are only applicable for stored video. It is not applicable to a real-time video application, since in a real-time video application, the needed information is not available before-hand.

The works of Hang and Chen [73, 74] reduced both the coding system and image signal into components of known mathematical models and then combined them together to form a complete video description. The theoretical foundation of this approach is the rate-distortion theory. Their contributions can be written as follows; firstly, they combined and extended the existing results to the standard-type video coders, and secondly, the non-ideal factors in real signals and systems are incorporated as adjustable parameters to compensate for their bias effects on the ideal model. They also used an analytic bits model (source model) to adjust the QP value and frame rate of a video coder to produce nearly constant quality pictures while limiting the motion jerkiness to within a tolerable range. They select a QP value only once for the entire image. Therefore, they claimed that their approach is much simpler than the other optimization formulations.

The study in [75], found that quantized transform coefficients, denoted by  $\rho$ , has a critical effect on the coding bit rate  $R$ , especially at low bit rates. They concluded that mathematically,  $R$  and  $D$  are also functions of  $\rho$ . The authors also found that two rate curves, called characteristic rate curves, which have interesting properties in the  $\rho$  domain. Based on these properties, they showed that the two rate curves can be directly estimated from the distribution of the transform coefficients. Then, they introduced a new concept of rate-curve decomposition to model the coding algorithm. The actual rate curve in the  $\rho$  domain is represented by a linear combination of the two characteristic rate curves. Based on the steps, a unified source model is developed for different video coding systems.

### **2.3.5.2 Quality Constrained Rate Control**

As stated in [14], the main idea of the quality constrained rate control research category is to adjust the coding parameters to achieve maximum picture quality and ensure the buffer never underflows or overflows. Since the amount of information in compressed video sequences is inherently variable, normally a buffer is placed between the video encoder and the transmission channel to smooth out the rate variation. The buffer also dictates the amount of delay in transmission systems. Larger buffer corresponds to long end-to-end delay. In practical video coding applications, it is essential to consider the rate/buffer/delay constraints.

Thus, several rate control techniques have been proposed to address various constraints of video transmission application. Among the techniques are to produce constant quality among video frames, or at least to smooth the video frame quality fluctuation.

For instance, Xie and Zeng [76] proposed a sequence-based frame-level bit allocation framework. They employed a rate-complexity model which is capable of tracking the video profiles without look-ahead encoding. The goal of their sequence-based bit allocation is to maximize the overall video quality, while at the same time minimize or reduce the distortion variation across the whole sequence within the constraints of the given target bit rate, frame rate, and delay requirement. They employed only single-pass real-time encoding (without analyzing future frames, which contributes to less delay). They explained that at a specific encoding time, it is possible to develop a greedy suboptimal solution based on all available information of the frames that have already been encoded. In particular, instead of explicitly minimizing the distortion variation or the average distortion across the frames, they addressed the problem by exploring a global bit allocation model that aims to characterize the relationship between the appropriate amount of allocated bits and a coding complexity measure of a frame. They claimed that their proposed framework can achieve smoother

video quality with less quality flicker and motion jerkiness. They also claimed that their solutions are better than MPEG-4 Annex L frame-level rate control, in terms of average peak-signal-to-noise ratio and quality smoothness.

### **2.3.5.3 From Objective Optimization to Subjective Optimization**

Chen and Ngan [14] also revealed the increasing video rate control research toward measuring or producing a coding algorithm based on video quality from the aspect of subjective evaluation. They highlighted several works heading in this direction. Among them are as follows; Chen et al. [77] with their work, “A unified framework of unsupervised subjective optimized bit allocation for multiple video object coding”, Lee et al. [78] with their work, “Foveated video compression with optimal rate control”, Itti et al. [79] with their work, “A model of saliency-based visual attention for rapid scene analysis”, and Tang et al. [80] with their work, “Visual sensitivity guided bit allocation for video coding”.

For instance, Tang et al. [80] claimed that in the usage of a psycho visual model for video rate control, visual attention (or foreground/ background analysis) is not the most important cue for proper bit allocation. However, they claimed further that the capability for human vision to detect distortion in video sequences is what a high quality video coder should take advantage of during the bit-allocation process. VDS is influenced by the motion structure as well as the texture structure of the scene. Thus, they proposed an effective visual distortion sensitivity model to indicate the perceptually important regions, which is a technique to evaluate the perceptual distortion sensitivity on a macroblock basis, and allocates fewer bits to regions permitting large perceptual distortions for rate reduction. The proposed algorithm can be incorporated into existing video coding rate control schemes to achieve the same visual quality at reduced bitrate.

#### **2.3.5.4 Hybrid Structure-Based Optimization**

The researchers in this group exploit the hybrid mode of optimization, which may involve a combination of different frame types (I-, P-, and B- frames), different coding modes (Intra, Inter, skip, etc.), different granularities of video (sequence, GoP, frame, etc.), and others.

Work by Song et al. [81], for example, proposed the optimal frame-skipping based on spatial and temporal tradeoff in the R-D sense. Both the frame rate and bit allocation were considered in the rate control framework. Their method is called variable-encoding frame-rate method, and its main purpose is for low-bit-rate video applications, particularly for H.263+. They used the mean of the Histogram of Difference (HOD) image values of current sub-GoP to decide the frame-rate of next sub-GoP. For example, if the mean HOD is larger than a certain threshold, it means that the frame rate could be decreased by one level. In this case, the bits assigned for each coded frame will be higher such that the frame distortion will be smaller. Moreover, a multidimensional rate control algorithm would address how to jointly control the frame size, frame rate, and QPs.

## **2.4 Network Transmission Protocol**

Presently, two of the most popular protocol used over the Internet are TCP and UDP [82]. Conventional wisdom believes that UDP is a better transport protocol than TCP for the video application, as verified in [29, 35, 83, 84, 85, 86]. This conclusion is readily accepted because UDP is a best-effort delivery service. Theoretically, there will be less delay and it provides better bit rates (throughput). Unfortunately, this best-effort transport protocol is potentially impeded by the performance of other applications that employ TCP, or worse, it would endanger the stability of the Internet [87]. Consequently, it potentially causes congestion-collapsed TCP protocol [88].

In addition to that, some organizations block these types of applications [89, 5]. Apparently because of those reasons, many studies had found that TCP is a more popular transport protocol than UDP [5, 86]. However, TCP is complex and dynamic, it employs congestion control schemes that are adapted dynamically to network conditions, and it retransmits loss or time-out packets. As a result, it often yields variable transmission rates and packet delays [83]. Its reliability has been inherently unsuitable for video application, which is a time-sensitive application [52].

#### **2.4.1 TCP-Based Video Transmission**

Several authors have stated their arguments on the high promise potential of video application over TCP transport protocol, as pointed out in [29, 85, 5, 90, 86, 91, 92, 93]. These researchers found that the bandwidth is adequate to access video application for many broadband users by using direct TCP [86]. Direct TCP means that without further improvement to the application technology or the protocols, it can still be run smoothly. For example, the study by Brosh and Baset [29] revealed that TCP may not be as delay-unfriendly as it is conventionally believed. The authors justified their finding by giving a reason on the congestion control mechanism used by TCP, which regulates the rate as a function of the number of packets sent by the application. Consequently, it is biased toward flows with small packet sizes. Since video applications use constant and small packet sizes [94], it has more advantage than applications with longer packet sizes. In addition to that, the studies by Wang et al. [86] also found that direct TCP video application generally provides good performance when the available bandwidth (achievable TCP throughput) is roughly twice the size of the video bit rates. Other studies found that streaming video clips on the Internet today are encoded at bit rates 89-300 Kbps [95, 96]. Also, we have witnessed the rapid deployment of broadband connectivity, which supports download rates of 750 Kbps – 1 Mbps [86]. These studies have also been supported by Boyden et al. [85]. They

revealed that streaming application is able to be transported via present TCP with good performance in a wide variety of scenarios, especially when the available bandwidth is more than the video bit rates.

## **2.4.2 TCP with Adaptive Bit Rate Transmission**

Perhaps the most major technical challenge in video transmission application is to adapt to the change of network conditions. Unfortunately, in the best-effort network, there is no way to select a unique target rate ahead of time. Should the bandwidth transmission rate degrade continuously, the media bit rate could not afford the same media quality, thus it leads to stalling in media display. In terms of user satisfaction, users would prefer continuous lower quality display than good quality display with intermittently pause.

The strategy of adaptive video application is to adjust the media rate according to network conditions, as mentioned in [97]. So that many commercial video application nowadays rely on multiple bit rates or adaptive bit rates to adapt to the changing network conditions. In practice, the range of adaptation should be discrete rather than continuous. The user perceive quality will suffer if there are frequent changes. Thus, another aspect to be considered is to maximize the range of supported rates and quality levels, and to provide the finest granularity of realizable points within that range. The greater the range and the finer the granularity, the more freedom there will be in the media rate-matching with the network bandwidth [97].

One of the methods of utilizing adaptive bit rates in response to changing network conditions is to use media scaling or content adaptation. According to Prangl et al. [89], media scaling is a method to reduce or increase the quality of media; for example, by changing the video frame rate, video resolution, different encoding layers, etc. Typical media scaling techniques are temporal scaling, quality scaling, and spatial scaling, as discussed previously.



### **2.4.3 UDP-Based Video Transmission Research**

In the UDP camp, the researchers are looking at the ways to make UDP video applications as TCP-friendly applications, in terms of fair bandwidth sharing with other flows, especially TCP flows. At the same time, they try to optimize the bit rate to fully utilize the available and permissible bandwidth quota. Based on the above premise, some research is being done to utilize UDP as a transport protocol for video applications while at the same time adding a rate control mechanism, as discussed by Apostolopoulos et al. [98].

Most of the research in this camp creates or modifies the UDP layer, or at the upper layer, so that it can be friendly to other flows, attempts to mimic TCP throughput, and avoids instantaneous fluctuations of TCP's AIMD algorithm [98]. These kinds of protocols introduce rate control or congestion control to the UDP, which are called TCP-friendly protocols. The protocol is said to be TCP-friendly when its bit rates do not exceed the maximum bit rates from a conformant TCP connection under equivalent network conditions, as mentioned in [87]. Similarly, Widmer et al. [99] defined TCP-friendly as when a flow does not reduce the long-term throughput of any coexistent TCP flow more than another TCP flow on the same path under the same network conditions.

Various methods had been studied and introduced to make UDP protocol suitable for video transmission applications. Some of the methods emulate, or even apply directly, TCP's AIMD behavior to be friendly to other TCP flows, as stated in [99]. Although they mimic the TCP behavior, the associated reliability mechanism is not included [99], and consequently it reduces delay. Therefore, the best of both worlds are gained, friendly to other TCP flows and at the same time, not creating unnecessary delay to the time-sensitive video application. Among other introduced mechanisms are Rate Adaptation Protocol (RAP) [88], Internet Friendly Transport Protocol (IFTP) [100], Loss-Delay Based Adaptation Algorithm (LDA+) [101], TCP Emulation At

Receiver (TEAR) [102], Random Listening Algorithm (RLA), Linear Proportional Response (LPR), and Nominee-Based Congestion Avoidance (NCA).

The RAP approach uses TCP's AIMD mechanism to be friendly to other flows. It does not implement full TCP congestion control but rather a partial implementation with less error (quality) or reliability control. Therefore, not all loss packets will be retransmitted. In specific, RAP uses loss-based rate control as an implicit feedback signal in the Internet due to the presence of fluctuating traffic [88]. They introduced a module, namely selective reliability, in which the module tries to deliver a maximum number of layers that can be fitted in an available bandwidth, while at the same time maintain the friendliness. IFTP can be said to be the extension of RAP, albeit as a result by different researchers. It uses not only AIMD to control the congestion while at the same time to be friendly to other flows, researchers stated that they mimic TCP in all of its stages, as reported in [100]. By doing so, they claimed that IFTP is superior to RAP in terms of suitability for both high and low loss rates (RAP fixes well in low loss rates only). They also embedded a simple scale factor extension to enable the protocol to adapt to the needs of more bandwidth with other concurrent TCP flows. Thus it is more capable of providing better QoS-differentiated services in order to select the best-effort flows [100]. They claimed that the protocol is better (compared to RAP) because their simulation not only tested TCP-friendliness, but also other important metrics such as packet delay, delay jitter, packet loss, and link utilization.

#### **2.4.4 Alternative Protocol for Video Application**

Other promising transport protocol standards for video application or real-time data are TCP-Friendly Rate Control (TFRC) [103] and Datagram Congestion Control Protocol (DCCP) [104]. They have been created by the individuals who have been involved in the development of other standards of popular transport protocols; protocol usability, and the stability of the Internet. As was written previously, UDP is not a favorable

protocol for video application due to unresponsiveness or unfriendliness to the other TCP flows. On the other hand, TCP inherits over responsiveness, creating unnecessary delay to the unneeded packet retransmission of the video application [82].

DCCP creation is motivated by the need for a new transport protocol that offers a reliable delivery; it accommodates alternative congestion control algorithms; accommodates the use of explicit congestion notification (ECN) [105]; and requires minimal overhead in packet size and CPU processing at the data endpoints. Moreover, only a minimal, but very much similar TCP's congestion control has been added to this protocol.

DCCP promises a stable, applicable and future-proof protocol that can be used to replace UDP. In addition, it was designed to accommodate alternative modular congestion control mechanisms, namely TCP-like, TFRC, and future, extensible or further form congestion control. By designing like that, it offers any application to negotiate suitable congestion control without the need to embed them in the application layer.

Regrettably, the industries are still slow in adopting this protocol. It is still rare to see this protocol in action and to gain the benefit of this new protocol. Further studies need to be done to incorporate advancements in previous video transmission studies relating to this protocol. The Internet community needs to be convinced on the benefit of this promising protocol; that is well equipped with congestion control but less reliability, promising stability and many other features. It is a good potential replacement for UDP (early transport protocol for realtime applications) and may be a better alternative for TCP video applications (which is too 'heavy' for real-time transport protocols).

While, the choice of TFRC for this study is discussed in the next chapter.

## **2.5 Related Work on Video Rate Shaping Algorithm**

Traffic shaping usually refers to a means that controls the volume of traffic to be injected into a network in a certain unit of time, or the maximum rate at which the traffic is sent (rate limiting), or a more complex criteria such as Generic Cell Rate Algorithm (GCRA) [106]. This control can be accomplished in many ways and for various purposes, for instance, a leaky bucket algorithm and token bucket algorithm. Traffic shaping is commonly applied at the end-to-end network to control traffic entering the network, but it can also be applied to the traffic source, as in the case of video transmission applications. Traffic policing is the distinct but related practice of packet dropping and packet marking. Therefore, rate shaping can be said to be policing the traffic admission into a limited bandwidth network.

Issues related to shaping video rate, in order to control its burstiness and produce streams that are more suitable for transmission, were initially started with great interest for the deployment in switched ATM networks, as discussed in [70]. This is because the switched ATM networks can provide some level of bandwidth guarantee, and video rate shaping is needed in shaping/policing the input to the limited bandwidth network. Typically, video rate shaping research in switched ATM networks is conducted in order to determine the requirement of the ATM network resources and traffic contracts for various video categories (action movie, sports, talking head, etc.) when video needs to be transported over an ATM network.

Among the work on video rate shaping in the ATM network can be found in [107]. In that work, the authors presented the impact of traffic shaping on enhancing the interactive video quality over the ATM networks. The effects of the leaky bucket traffic shaper on video quality was studied under different token buffer rates and token buffer sizes, and the tradeoffs involved were discussed. An enhanced traffic shaper using the early drop mechanism was then introduced. Simulation results showed that for delay sensitive applications like interactive video, the combination of traffic shaping

and the early drop mechanism had improved video quality significantly compared to VBR video and regular traffic shaping.

Work in [108] used a rate shaping algorithm based on a selective information discard approach applied to MPEG-2 Variable Bit Rate (VBR) sources, which tests several output constrained bit rates in order to match the channel bandwidth requirements. They introduced Dynamic Rate Shaping (DRS), thus obtaining a greater flexibility in the choice of transmission parameters. In fact, according to the authors, it is possible to trim the required bandwidth in continuous mode in the working range, thus saving some bandwidth when keeping the video quality constant during changing fade conditions. The performances of DRS applied to both the non-scalable and the scalable MPEG-2 coded video streams were shown separately.

Most of the research works of video rate shaping, especially early works, implement the rate shaping in stored-video system or other kinds of video application, which can tolerate certain duration of delay. However, there are certain research works of video rate shaping, which are implemented for a real-time video application. They will be discussed later in Subsection 2.5.3. Besides that, it is very common that video rate shaping uses certain traffic policing tools, such as a leaky-bucket algorithm, token-bucket algorithm, and etc.

### **2.5.1 Leaky-Bucket Algorithm**

The Leaky Bucket Algorithm is based on an analogy of a bucket that has a hole in the bottom through which any water it contains will leak away at a constant rate until or unless it is empty [109]. Water can be added intermittently, i.e. in bursts, but if too much is added at once, or it is added at too high an average rate, the water will exceed the capacity of the bucket, which will overflow.

In computer network research, the leaky bucket is an algorithm employed to ensure that data transmissions conform to defined limits on bandwidth and burstiness. It has

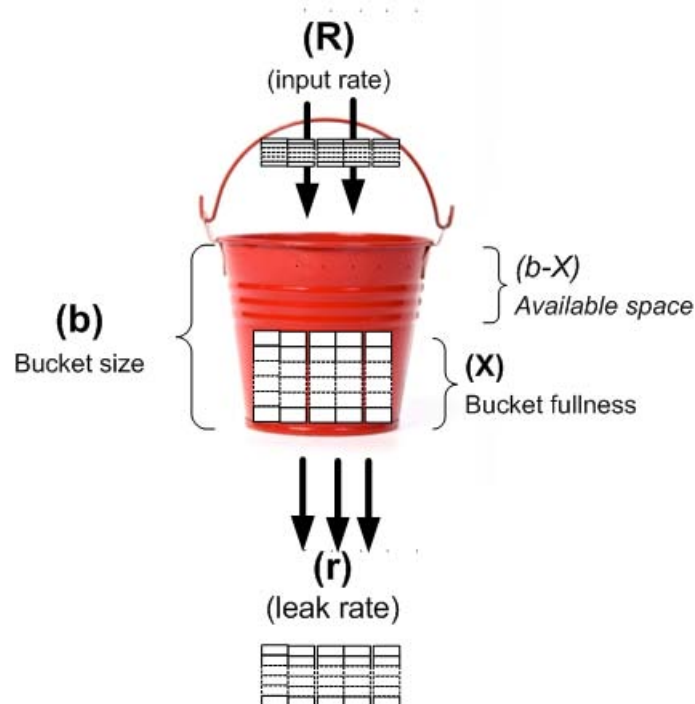
been used widely in policing the data traffic into the network. The QoS of the network can be maintained if the traffic in each connection is monitored and regulated by a “policing function” or “usage parameter control”, as stated in [110]. Policing functions enforce and ensure that the traffic from each user complies with the negotiated usage parameters, in order to efficiently monitor the arriving traffic and respond to any violation of usage parameters.

The leaky-bucket algorithm is implemented by manipulating two main parameters; leak rate,  $r$ , and bucket size,  $b$ , and it can be noted as  $LB(r, b)$ . In network transmission, the bucket size can represent a network interface buffer, and the leak rate can constitute the network transmission rate. Other typical parameters are the data input rate,  $R$ , and the bucket fullness level,  $X$ . The data input rate will represent the application data that potentially enters the network, and the bucket fullness level can be used as an indication for the network rate shaper controller – either the network is under-utilized (underflow) or over-utilized (overflow). Thus,  $R$  will be controlled by the rate shaper engine to ensure that it utilizes the network at an optimum level. Figure 2.6 illustrates all the above-mentioned parameters in a leaky-bucket system.

### **2.5.2 Video Rate Shaping Research in Stored-Video Application**

As mentioned previously, the stored-video application refers to video transmission which is not transmitted in real-time. These are such as delayed-telecast, Video on Demand (VoD), and etc. Most of the early works in video rate shaping area are stored-video based application.

Among the works which fall into this category is the work done by Alam et al. [111]. They proposed a novel traffic shaper of IP packets at the MPEG video source. In their work, they discussed the effect of the traffic shaper parameters on delay and buffer requirements. They verified their traffic shaping model by comparing the delay and buffer requirements obtained from simulation of several different MPEG video



**Figure 2.6:** Parameters in a leaky-bucket system

sequences. The simulation results showed excellent agreement with the theoretical model. The relationship between the traffic shaper parameters and traffic specifications are also studied.

Depending on the shaper parameters chosen, such leaky-bucket or constant-bit-rate traffic shapers either fail to utilize the full burst-handling capability of Guaranteed Service (GS), or cannot ensure that the IP packets sent out to the Internet is conformant to the pre-negotiated traffic specifications. A leaky-bucket traffic shaper reduces or removes the burstiness of the MPEG data stream, and may introduce a large amount of delay, which can result in a large buffer requirement at the traffic shaper, as discussed in [112].

### **2.5.3 Video Rate Shaping Research in Real-Time Video Application**

In the case of real-time VBR encoding, unless the coding and rate control parameters are selected very carefully, the quality of video may vary significantly over a single video session. This is because the nature of real-time coding limits the bit rate optimization process since information about video complexity is limited to the current and past frames, and no information exists about future frames.

Among the earliest works, which is working on real-time video rate shaping, is the work done by Gringeri et al. in [70] (this fact was claimed by the authors, however in the literature the first work by Hamdi et al. [23] is chronologically earlier). Gringeri et al.'s work primarily was on shaping existing streams before they enter the network to reduce their burstiness and optimize utilization of network resources. That technique is well suited for MPEG streams that have already been encoded. In their work, a scheduling technique was presented, which selects a traffic contract for a pre-encoded MPEG video stream with the criteria of minimizing network resources and maintaining video quality. The network resources required to transmit a stored variable-rate MPEG can be reduced by properly analyzing and smoothing the video stream before transmission.

They claimed, by using an analysis method for a pre-encoded video stream such as the just-in-time scheduling method, a traffic contract can be found that guarantees video buffer verifier (VBV) compliance. This traffic contract and an effective bandwidth metric can be used to measure the potential savings possible when that video clip is transmitted over an ATM network. These savings are based on simply shaping the output of existing variable-rate encoders used in storage applications. The burstiness of MPEG streams varies greatly depending on the category of the video being encoded. For example, low-motion talking head clips are significantly less bursty than sports or action movie clips. This implies that different program



materials will require network resources based on the complexity and motion of the video being encoded. Therefore, understanding the characteristics and number of multiplexed video streams is important since it allows the effective bandwidth metric to be configured to provide more accurate results.

Other works used simpler ways to calculate rate control, which is usually done at the Group of Picture (GoP) level. For example, the work in [23] had manipulated a leaky-bucket algorithm in its Shaped VBR (SVBR) to determine a suitable bit rate for the next GoP video to be transmitted. This algorithm will be described in the next section. Since the introduction of SVBR, it has been utilized by several works. However, these works still implemented under the limitation of the SVBR. These will be discussed further in the next subsection.

#### **2.5.4 Recent Work on Video Rate Shaping**

In recent research, work by Semsarzadeh et al. as in [9], the researchers claimed that current encoders have no strategy for acting in response to bitrate variations that occur as a result of changes in network condition. The time it takes to adapt has a significant impact on the received video quality. A slower response means that more packets are dropped by the network, which in turn results in a lower video quality. Furthermore, a slower response makes the congestion persist for a longer period of time.

They claimed that the present rate control H.264/AVC, the state of the art video compression standard, calculates the new QP as a weighted average of all the previous QPs. If network capacity changes during a session the last wanted bit ( $W_n$ ) changes accordingly, which in turn results in a new  $Q_n + 1$  value. Since averaging is performed over all coded frames, changes in  $Q_n + 1$  are insignificant, making the reaction to network conditions very slow. A slow bit shaping usually results in quality degradation due to packet loss.

The researchers suggested that in order to speed up the bitrate adaptation, the QP change has to adapt to the current QP values by assigning more weight to the latest wanted bitrates than the former ones. In other words, each QP has to be less dependent to previous frames. One solution is to limit this formula to the current GoP.

Another recent work can be found in [113], where the author claimed that conventional rate policing such as a generic cell rate algorithm is inadequate to sufficiently regulate the transmission of VBR data sources over bandwidth limited by ZigBee. IEEE 802.15.4 ZigBee is a standard designed to address the needs of low-cost, low-power, wireless sensor networks for remote monitoring, home control, and building automation network applications in the industrial and domestic markets. Therefore, it is impossible to transmit MPEG VBR video over the ZigBee channel. A buffer entitled 'traffic-shaping buffer' is introduced to prevent excessive overflow of MPEG video over the ZigBee channel. A new Neural-Fuzzy (NF) scheme is developed to adjust the traffic-shaping buffer output rate and eliminate unacceptable delay or loss of the VBR encoded video and to conform with the data to the token-bucket's contract prior entering the ZigBee channel.

A Rule-Based Fuzzy (RBF) scheme is developed to monitor the video rate entering the traffic-shaper, in order to prevent either saturation or starvation of the buffer. The second control scheme is the NF controller, which observes and reduces the burstiness of the departure rate from the traffic shaping buffer to enable a smooth transmission of MPEG VBR video over ZigBee while satisfying two objectives, namely to reduce traffic congestion in the network and to maintain image quality.

Although the researchers claimed that the produced system does not require great processing power, reduction in data loss, and time delay, which could be very useful for time-sensitive MPEG VBR video services, the problem with this solution is the usage of a traffic-shaping buffer. The usage of additional buffer generates delay to the system, which is unsuitable for time-constraint video applications.

## **2.5.5 Shaped Variable Bit Rate (SVBR)**

Shaped Variable Bit Rate (SVBR) is a novel video rate shaping for real-time video transmission applications. It was created by Hamdi et al, as discussed in [23]. While the detailed analysis and the empirical studies in the purpose of evaluating its strengths and weaknesses will be comprehensively discussed in Chapter 4, this section briefly discusses the SVBR algorithm and then reviews the recent work.

### **2.5.5.1 Shaped Variable Bit Rate (SVBR)**

SVBR can be regarded as an alternative solution that takes advantage of both CBR and VBR, while at the same time, it eliminates the weaknesses of both rate controllers. It encodes the video with an open loop VBR as much as possible, and at the same time it controls traffic admission into the network when it exceeds a permissible level. Thus, it maintains a consistent visual quality in VBR and gets into a reasonable compromise with adaptive quality when the network transmission rate degrades. At the same time, it avoids unpredictable large bursty rate variations, as in VBR, but without the rigidity and systematic coding delay of CBR coders [51].

SVBR algorithm limits the open-loop burst while at the same time allows for open-loop VBR coding, provided, they are still within a permitted constraint. To achieve that function, SVBR manipulates a leaky bucket algorithm in performing an admission control. The leaky bucket used by Hamdi et al. [23] can be considered as an imaginary buffer, thus no extra delay is introduced. Moreover, they assumed that for a fast moving scene with complex image structures, the algorithm can reduce slightly the scene quality since human eyes do not have enough time to notice the image details. In addition, they have suggested applying the algorithm at GoP granularity, consequently, this will yield to a less complex algorithm and lower delay.

However, despite its novel creation for real-time applications, the analytical empirical evaluation to SVBR algorithm, as described in Chapter 4, found some

obvious weaknesses. The weaknesses that have been strongly stressed are the occurrence of a sharp decrease in the video rate, the occurrence of a bucket overflow, the existence of a low video rate with a low bucket fullness level, and the generation of a cyclical negative fluctuation. As elaborated in Chapter 5, one of the fundamental contributors of the above weaknesses is as a consequence of the estimation and prediction used in generating the SVBR video rate.

#### **2.5.5.2 Recent SVBR Related Work**

Since the introduction of SVBR, it has been utilized by several works. However, these works were still implemented under the limitations or weaknesses of the SVBR.

#### **Evalvid-RA and RA-SVBR**

One of the recent works was performed by Lie and Klaue, as in [2, 7]. They created the Evalvid-RA, a tool set for rate adaptive video performance evaluation in ns-2. The creation of Evalvid-RA builds on modifications to the original software of Klaue et al. [114] and Ke et al. [24]. The main modification to EvalVid described in Subsection 2.6.4) was that the re-assembly post-process program had to take into account multiple MPEG-4 source files and modification to the ns-2 interface, and the associated VBR rate controller based on SVBR by Hamdi et al. Thus, the engine of the video rate adaptation in the Evalvid-RA comes from the SVBR algorithm.

They claimed that Evalvid-RA is a true rate adaptive video. Their solution has generated a real rate adaptive MPEG-4 streaming traffic, using the QP for adjusting the sending rate. Then, a feedback based on the VBR rate controller is used at simulation time, which supports TFRC and a proprietary congestion control system named P-AQM. By simulating TFRC and P-AQM in ns-2, they demonstrated the Evalvid-RA capabilities in performing close-to-true rate adaptive codec operation with low complexity.

They also claimed that their algorithm is based on the SVBR algorithm with several enhancements. Among the enhancements they made, besides implementing SVBR in Evalvid framework to become Evalvid-RA, are that they added a supported network feedback system and performed some changes in the parameters of the SVBR's leaky-bucket algorithm. Instead of using SVBR leaky-bucket equation, as in Equation 2.1, they used network feedback and reformulate the equation as Equation 2.2,

$$X(k+1) = \min \{b, (\max \{0, X(k) - r\} + R(k))\} \quad (2.1)$$

$$X(k+1) = \min \{b', (\max \{0, X(k) - r'\} + R(k))\} \quad (2.2)$$

where  $r' < r$ ,  $b' < b$ , where  $r'$  and  $b'$  are dynamically adjusted based on network feedback, i.e.  $r' = r_{old}i/G + r_{new}(G-i)/G$  and  $b' = b_{old}i/G + b_{new}(G-i)/G$ . When there is no network feedback during a GoP,  $r' = r_{old} = r_{new}$  and  $b' = b_{old} = b_{new}$ .

What should be stressed here is that although the changes made in Evalvid-RA seems big, it still maintains the core SVBR algorithm. By limiting the changes to SVBR leaky-bucket parameters, such as  $r' < r$  and  $b' < b$ , all the weaknesses in SVBR are inherited.

### Other Recent SVBR Related Works

Another research by Talaat et al. as in [31] investigated the effect of incorporating TFRC on the peak signal-to-noise ratio PSNR of the transmitted video over the Internet in a simulated environment. They found that TFRC performance on slow motion videos was slightly better than on medium-motion that was better than high-motion videos. In this work, they actually deployed the Evalvid-RA in their investigation, without making any changes to the core SVBR algorithm.

Another work can be found in [115], where a power management mechanism for wireless video transmission using the TFRC protocol takes into account feedback about the received video quality and tries to intelligently adapt transmitting power

accordingly. The purpose of this mechanism is to utilize TFRC feedback and thus achieve a beneficial balance between power consumption and received video quality.

The researchers claimed that they had implemented a module consisting of the logic of the proposed mechanism in the Evalvid-RA environment. The module that implemented the TFRC protocol also was changed so that, they claimed, it can provide information about packet losses to their mechanism. The mechanism calculates the power needed to improve PSNR, and then this information was passed to the modified wireless physical layer module that is able to increase or decrease power according to the mechanism. However, as stated previously, the fundamental contributors for the weaknesses in SVBR/Evalvid-RA is as a consequence of the estimation and prediction used in generating the video rate, the work in [115] did not change anything on this part.

Further work was done by Bouras et al. in [116, 117, 118], where they performed a performance evaluation of MPEG-4 video transmission with their proposed multicast protocols, namely Adaptive Smooth Simulcast Protocol (ASSP) and Adaptive Smooth Multicast Protocol (ASMP). The features in their protocols include adaptive scalability to large sets of receivers, TCP-friendly behavior, high bandwidth utilization, and smooth transmission rates which are suitable for multimedia applications. They evaluated the performance of their protocols under an integrated simulation environment which extends Evalvid-RA to the multicast domain with the use of the Real-time Transport Protocol (RTP) or Real-Time Transport Control Protocol (RTCP) protocols. Simulations conducted under that environment combined the network-centric measurements along with video quality metrics. They claimed that the “joint” evaluation process provides a better understanding of the benefits and limitations of any proposed protocol for multimedia data transmission.

They called their tool set as Multi-Evalvid-RA, which provides all the necessary tools to perform simulation studies and assesses the video quality by using both

network related metrics along with video quality measurements. This is due to the fact that Evalvid-RA does not support multicast transmissions, which is necessary for experiments and simulations with the RTP/RTCP protocols. Therefore, they further extended the functionality of Evalvid-RA by adding the codes in order to exploit the sender and receiver RTCP reports.

They used Evalvid-RA in implementing media rate control based on traffic feedback and took advantage of RTCP's Sender Report (SR) and Receiver Report (RR). They claimed that the innovation created is the calculation of smooth transmission rates, which was performed by receivers and based on RTCP reports, which resulted in the oscillations being reduced. Another important attribute is the long term TCP-friendliness. Although, this work is dissimilar from others in terms of implementing the SVBR in a multicast environment instead of unicast and making some adjustment in video rate, it did not make any adjustment to the estimation approach in the SVBR.

## **2.6 Video Performance Evaluation**

As mentioned before, video transmission has attracted much interest among researchers. Therefore, being able to measure its quality has significant importance at all stages of video research. As mentioned in [119], the measurement started from the development of new video codecs, to the monitoring of the transmission system's quality of service, and including the transmitted video quality (i.e either the distortion or artifact is able to be minimized). This became more apparent with the advent of compressed digital video, because digital video exhibits artifacts fundamentally different from analog systems – blockiness, blurring, motion estimation mismatches, and etc. Furthermore, transmission over networks that do not provide strong QoS support introduces additional types of distortion (e.g., artifacts due to packet loss).

Most of the previous research on video transmission used network performance metrics, namely less delays/jitters, less packet loss, higher bandwidth utilization/throughput, and etc. to evaluate the result [24]. Although those metrics certainly influence video quality, however, they are not be easily or uniquely transformed to reflect the quality of video transmission. For instance, Ke et al. [37] stated the following example, a 3% packet loss could be translated into a 30% frame-error probability. Furthermore, modern video codecs are hierarchical, so the loss of the I-frame would cause other frames in the same GoP to become useless. According to [114] the transformation could be different for every coding scheme, loss concealment scheme, and delay/jitter handling. Consequently, network performance metrics are considered not sufficient to measure the user perceived quality adequately [24, 2, 114, 120] and the user perceived quality impression or observation is, nevertheless, the most important factor. This is because the quality of a video transmission depends on the impression of a human observation of the delivered video, as stated in [7].

Miras [119] stated that although the quality is a rather difficult concept to grasp, two features of quality measurement have been widely accepted. Firstly, quality implies a comparison, and secondly, quality must be measured in some open ended scale, which means that the quality of the test video sequence is allowed to be both better or worse than the reference. A direct comparison takes place when both the original (or reference) and the impaired (or test) video sequences are available and shown to the viewer or passed to the measurement instrumentation. When the reference is not available, then comparison (in the case of subjective tests) is done using some “internal” reference of quality of the individual viewer. Therefore, video quality measurement can be categorized into several classifications.



### 2.6.1 Video Performance Evaluation Categorization

As mentioned in the previous subsection, the researchers in video transmission studies either evaluate the result in terms of network performance metrics or performance metrics that are related to the video. The former metrics or namely conventional network performance metrics, among others are the utilization of networking resources, delay, jitter, buffer occupancies, and buffer overflow probabilities. All are related to the networks that carry video traffic. The latter metrics, grouped as the starvation probability, is the quality of the video itself, as mentioned in [4]. These two mentioned-metrics give some indication of the quality of video delivered to the user over the network under study.

According to Seeling et al. [4], starvation probability or starvation loss probability comes in two main forms. The first form is the frame starvation probability, which refers to video frames that miss their decoding (playout) deadline. The frames decoding deadline are those frames that are not completely delivered to the receiver by the time the receiver needs them to start the decoding. The second form is the information loss probability, which refers to the distortion of encoding bits. The information loss probability has a finer granularity than the frame loss probability because in frame loss probability a partially delivered frame is considered as one lost frame toward the frame loss probability (irrespective of how much of the frame was delivered/not delivered in time), whereas the information loss probability counts only the fraction of the frame's information bits that were not delivered in time.

Those above-mentioned premises certainly have some impact on the type of video frame involved. For instance, the I-frame, which is a reference frame for the following frames in a GoP, the loss of it is essentially equivalent to the loss of all the frames in the GoP. Similarly, a loss of given P-frame, which is required to decode all the successive P-frames in the same GoP as well as the B-frames encoded with respect to these P-frames, is equivalent to the loss of all these dependent frames.

Meanwhile, the information loss probability is mainly motivated by error concealment and error resilience techniques that allow for the decoding of partially received video frames. Error resilience techniques are currently a subject of intense research efforts, and more advances in this area are to be expected. The deployment of these techniques may be affected by the required computational effort and energy. Therefore, this research is not focusing on this type of starvation loss probability. Frame lost probability is more related to this study.

The frame loss probability and information loss probability are convenient performance metrics for video networking as they can be directly obtained from network simulation with video traces. However, these loss probabilities only provide limited insight into the video quality perceived by the user. It is certainly true that a smaller loss probability corresponds in general to a higher video quality, but it is difficult to quantify this relationship.

The above mentioned paragraph has led to another categorization of video quality assessment. This categorization is known as subjective video assessment and objective video assessment. According to Miras [119], subjective quality assessment is aimed to capture the user's perception, and inevitably produce some form of rating or quality score that corresponds to the viewer's judgement of quality. In this method, a panel of human subjects is shown a series of video sequences, and asked to score the quality of video scenes. Depending on what contextual factors that influence user perception need to be derived, three testing procedures are most commonly used: Double Stimulus Continuous Quality Scale, Double Stimulus Impairment Scale, and Single Stimulus Continuous Quality Evaluation.

Although subjective quality testing procedures constitute a reliable method for gaining insightful knowledge about the performance of a digital video system, it is complicated and costly, which make these methods unattractive and not feasible for automating the quality evaluation process. The involvement of human subjects renders

this approach unusable when the quality monitoring system has to be embedded into practical processing systems. Furthermore, subjective tests are very sensitive to viewing conditions and the number of human subjects required to draw statistically sound and safe results.

This is because the operating characteristics of digital transmission systems like bandwidth, packet loss, bit-error rate, and delay may change over time producing quality fluctuations. These transients may be very difficult to capture, unless the performance of the system is being continuously monitored. Therefore, only continuous and non-intrusive performance monitoring can accurately capture what the viewer is perceiving in these instances. Thus, objective video assessment is a good alternative to overcome those problems.

Objective video assessment does not use human subjects. Instead, it measures and analyzes the video signal for perceivable distortions and artifacts, usually by using some automation, such as by utilizing computers. The most traditional ways of evaluating quality of a digital video processing system are by employing the Signal-to-Noise Ratio (SNR) and Peak Signal-to-Noise Ratio (PSNR) between the original video signal and the transmitted one. PSNR is the most widely used objective video quality assessment metric.

Despite being a straightforward metrics to calculate, this measure operates strictly on a pixel-by-pixel basis, it neglects the type of distortion, the image content and the viewing conditions have on the actual visibility of artifacts. The main problem with PSNR is that it cannot differentiate between impairments that humans can and cannot see, or impairments that are less or more annoying. Recently, as stated in [121], a number of more complicated and precise metrics were developed, for example Video Quality Measurement (VQM), Perceptual Evaluation of Video Quality (PEVQ), and Czenakowski distance (CZD).

However, objective form of video quality assessment is very important in video transmission research. As discussed in [122], when estimating quality of a video codec, all the mentioned objective methods may require repeating post-encoding tests in order to determine the encoding parameters that satisfy a required level of visual quality, making them time consuming, complex, and impractical for implementation in real commercial applications. For this reason, much research had focused on developing novel objective evaluation methods, which enable prediction of the perceived quality level of the encoded video before the actual encoding is performed.

## **2.6.2 PSNR**

PSNR is considered as the simplest form for objectively measuring the video frame quality by calculating distortion at the pixel level. Due to their relative simplicity, fidelity metrics, the Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE) are extensively used by the image processing community, as discussed in [20]. According to Koul [123], PSNR is easy to compute and well understood by most researchers. It has been considered to be a reference benchmark for developing perceptual video quality metrics [124].

In an engineering term, PSNR is the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation [125]. PSNR is usually expressed in terms of the logarithmic decibel scale [126]. The signal in this case is the original data, and the noise is the error introduced by compression or transmission. When comparing compression codecs, it is used as an approximation to human perception of reconstruction quality, therefore in some cases one reconstruction may appear to be closer to the original than another, even though it has a lower PSNR (a higher PSNR would normally indicate that the reconstruction is of higher quality).

One has to be extremely careful with the range of validity of this metric. Many researchers gave caution on this aspect, this is because it is only conclusively valid when it is used to compare results from the same codec (or codec type) and same content, as mentioned in [124].

PSNR is then defined as (it is derived by setting the MSE),

$$PSNR = 10 \log_{10} \left( \frac{MAX^2}{MSE} \right) \quad (2.3)$$

where  $MAX$  is the maximum possible pixel value for each frame and MSE is defined as

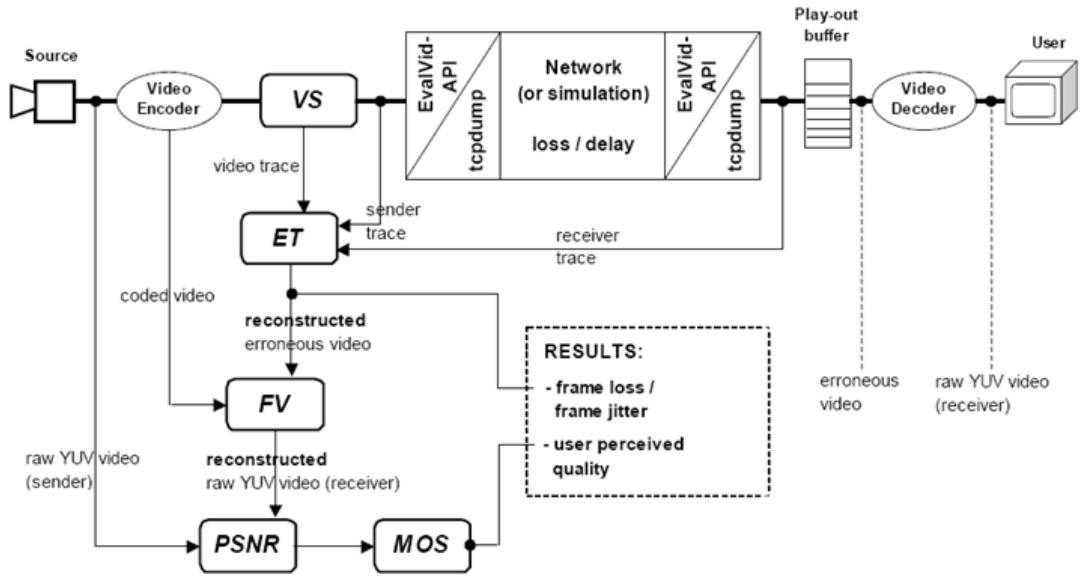
$$MSE = \frac{1}{m.n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2 \quad (2.4)$$

where  $m$  and  $n$  are the frame size in pixel,  $I(i, j)$  is the original signal at pixel  $(i, j)$ , and  $K(i, j)$  is the reconstructed signal at pixel  $(i, j)$ . Thus, every single pixel error contributes to the decrease of the PSNR, even if this error is not perceived. The result is a single number in decibels, ranging from 30 to 40 for medium to high quality video.

One of the popular tools for measuring video quality in terms of user-perceived quality in the simulated environment, particularly PSNR, is Evalvid. This tool set is going to be elaborated in the following subsections.

### 2.6.3 Evalvid

Although there are still many arguments on how to best evaluate the performance of video quality delivered in a simulated environment, many believe that user perceived quality is good enough, as mentioned in [2]. The growing concern on this issue has led to the introduction of Evalvid [114]. The Evalvid is a complete tool or framework to evaluate the quality of video transmission either in a real or simulated network. Figure 2.7 redraws the original framework of Evalvid.



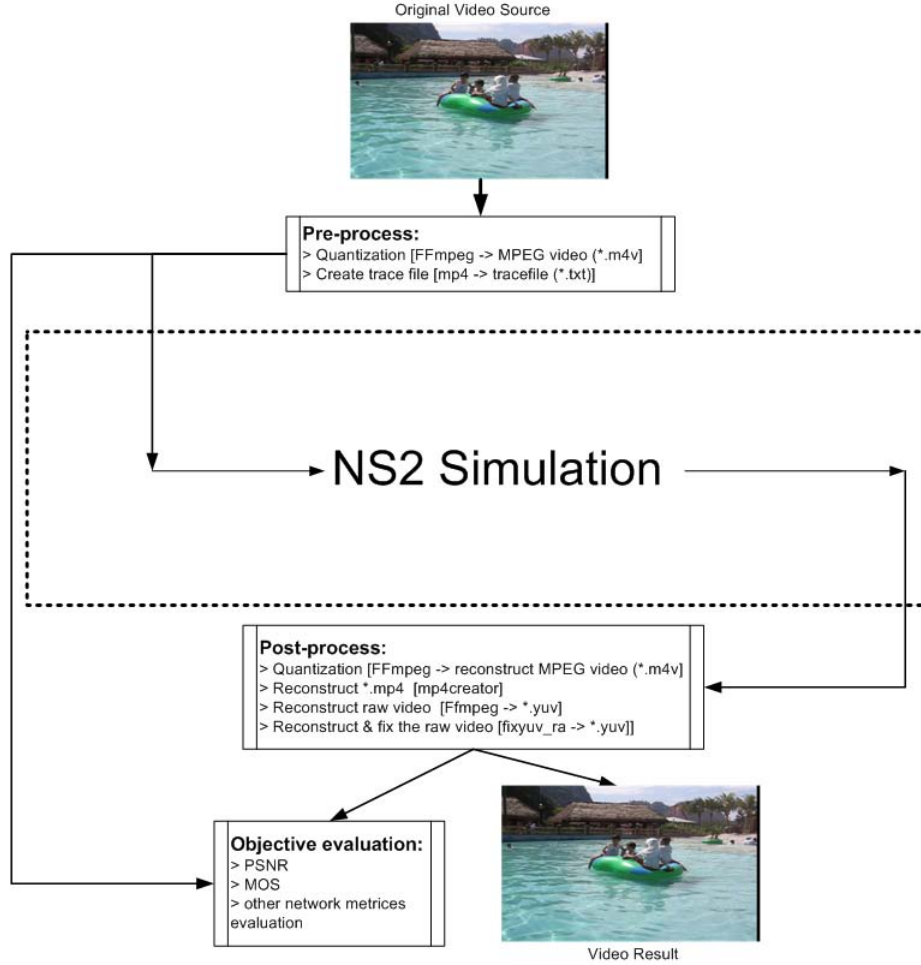
**Figure 2.7:** The original Evalvid framework

The Evalvid evaluation consists of network level and user perceived video metrics, such as PSNR, fraction of decodable frames, and MOS. Therefore, the researchers are able to evaluate their network designs or setups in terms of perceived video quality by the end user. Original Evalvid does not have network simulation agents to enable seamless integration of Evalvid and ns2. Thus, it enables the researchers to have greater freedom to analyze their proposed network designs for video transmission, for instance, to simulate real video streams over a large set of network scenarios, including relatively large topologies, node mobility, different kinds of concurrent traffic, or many other simulation objectives.

#### 2.6.4 Evalvid-ns2 Integration and Evalvid-RA

Ke et al. in [24] claimed that when attempting to use better user perceived quality, such as MOS and PSNR in a simulated network environment, it is difficult and sometimes almost impossible due to limited and different characteristics of publicly available video traffic traces. Thus, they integrated Evalvid with ns-2 to enable seamless video transmission evaluation in a ns-2 network simulation. They introduced several

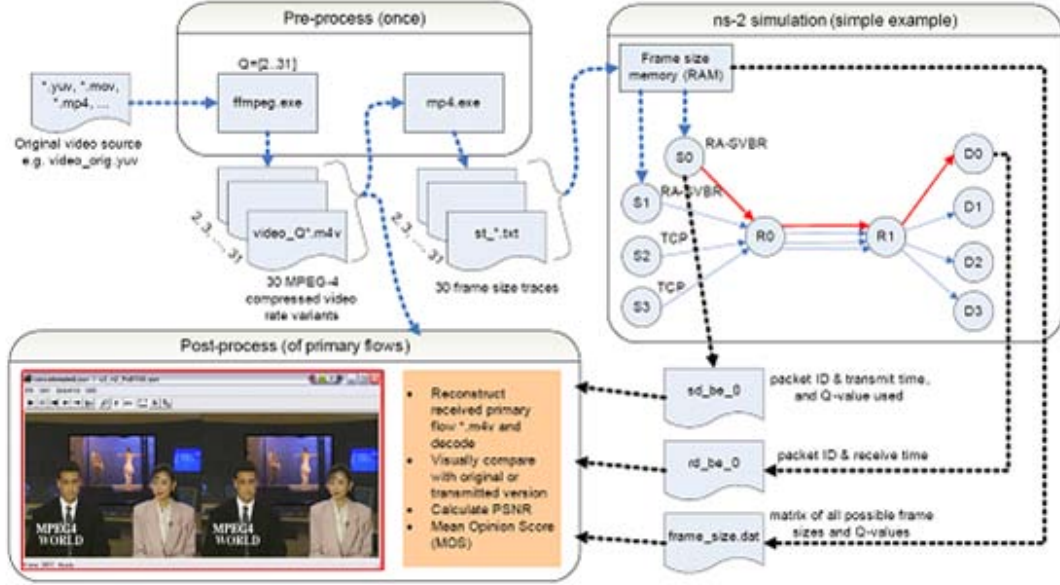
interface agents and some changes to the original Evalvid to enable the integration, and fix the inherent problems. By using EvalVid-ns2 [24], it enables researchers and practitioners in general to simulate and analyze the performance of real video streams with consideration for video semantics under a vast range of network scenarios. Figure 2.8 illustrates the Evalvid framework in a simulated environment.



**Figure 2.8:** Evalvid framework in simulated environment

Then, motivated by the lack of rate adaptive media content in the previous Evalvid solution, Arne Lie and Jirka Klaue in [2] introduced Evalvid-RA. The tool set is able to generate real rate adaptive video streaming traffic. Therefore, the solution will be friendly to other flows should it be implemented in the real-world network. Besides that, the authors also implemented this solution in DCCP-TFRC with a proprietary

congestion system control, namely P-AQM. Figure 2.9 illustrates the Evalvid-RA framework.



**Figure 2.9:** Evalvid-RA framework (redraws from [2])

## 2.7 Summary

This chapter began with a description of the challenges in transporting video over the Internet. Then, the elaborations were on the role of video rate control in the video transmission system, the relationship of video rate control and video coding, the relationship of QP and the rate control, the traditional rate controls, the Rate-Distortion (R-D) theory, and research in the video rate control. Finally, related works of the video rate shaping algorithm was analytically studied. This included the related research and progress in the video rate shaping algorithm, and the early and recent works in this research domain.

In real-time video rate shaping, researcher focus were toward the SVBR algorithm. It was highlighted that SVBR had been designed with great advantages for real-time video applications. However, it was pointed out that the algorithm inherited several



serious weaknesses. It has been stressed that since the introduction of SVBR, the algorithm had been utilized by several recent works. However, these works had not changed the core of the SVBR algorithm. In these works, the improved SVBR-based algorithms were implemented along with the limitations or weaknesses of the SVBR. From these background studies, the main issue arose is, what is the strengths and weaknesses of the SVBR algorithm? By understanding the reasons of its strengths and weaknesses, the design of a new rate shaping algorithm will be better produced.

In the next chapter, the methodology and simulation setting employed in this research work will be deliberated. The deliberations are on the methodology options, the challenges to fully grasp the behavior and performance issues of a protocol without evaluating it under controlled conditions, and the chosen way of performing the research performance evaluation.

# **CHAPTER THREE**

## **RESEARCH METHODOLOGY**

This chapter discusses the methodology used in carrying out the research, whereby its purpose is to present the approaches used in executing the research and to demonstrate how the approaches can deliver to meet the objectives of the research.

In presenting the purpose of this chapter, several subsections have been outlined. It starts by outlining the research steps and research methodology options available. Then, the selected approach is justified. In Section 3.3, the metrics in evaluating the performance of the new algorithm are explained. In Section 3.4, the details of the experiment setup are elaborated. In Section 3.5, discussion on how the experiments, design, and the result are verified and validated. Finally, the conclusion of this Chapter is summarized.

### **3.1 Research Operational Framework**

To accomplish the goal of this research, the following research steps were carried out:

#### **Phase One:**

- i. Performing an in-depth study on the video transmission rate control problems, rate control algorithm, video performance evaluation techniques, and the

surrounding issues.

- ii. Performing an in-depth study on the algorithm in SVBR. Then, execute an empirical evaluation to indentify the strengths and weaknesses of the algorithm. After that, identify the areas that should be eliminated or enhanced for video application.

**Phase Two:**

- iii. Designing a new shaped algorithm for video application. The design processes include determining of design requirements, specifications, justifications, and objectives for the new algorithm.

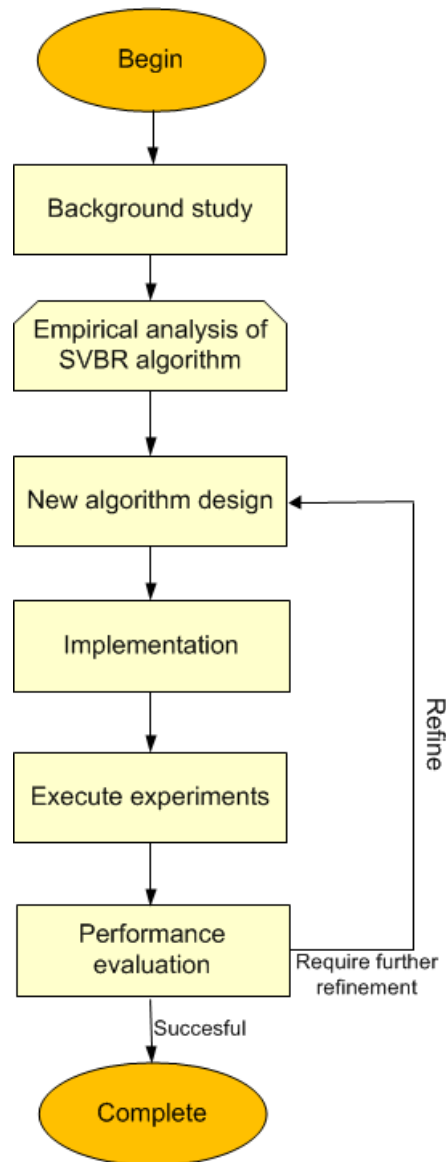
**Phase Three:**

- iv. Implementing, validating, and verifying the design of the new shaped algorithm. The algorithm to be implemented in Network Simulator 2 (ns-2).
- v. Executing the experiments with various video sequence activities.
- vi. Performing performance evaluation of the new algorithm by comparing it with the SVBR algorithm.

These research steps are illustrated in Figure 3.1. The steps have been categorized into three phases in order to achieve the three research objectives. The first phase is to fulfill the first objective, so as with the second and third objective.

In the first step, the study focuses on the issues surrounding the video data transmission and the rate control. This study provides sufficient background information and sets the context of this research. Moreover, the literature review that relates to this research is studied thoroughly and analytically.

The second step is to analyze and evaluate extensively the strengths and weaknesses of the Shaped VBR algorithm. The output of this step is a set of potential enhancement



**Figure 3.1:** Research steps

to the original SVBR algorithm, to be applied in a slight delay video transmission application.

After knowing the potential improvements of the original SVBR algorithm, the third step is to design a new algorithm. This step creatively proposes a new alternative of creating an algorithm with a high video rate but without bucket overflow, and at the same time it is able to obtain smoother Quantization Parameter (QP) fluctuation.

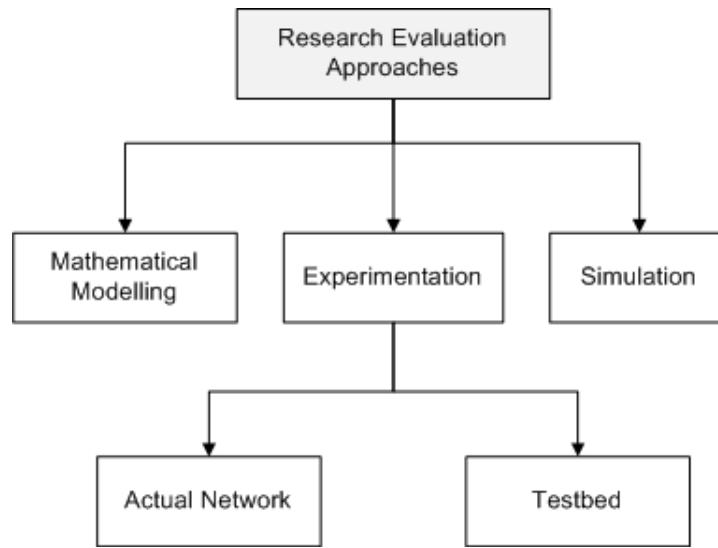
The fourth step is to implement the algorithm, so that it can be tested and evaluated in order to measure its performance. The implementation is done in Network Simulator 2 (ns-2). For the purpose of measuring the new algorithm performance using the user-perceived video quality metric, the design is implemented in the Evalvid-RA [2] environment package.

After the implementation, the design experiments set up, choosing appropriate parameters, and executing the experiment are done in step 5. The output from this step is a set of data for performance evaluation.

The last step is to evaluate the new algorithm performance. This evaluation is done by comparing it with the SVBR algorithm. This step is needed when researchers want to compare a number of alternative designs and find the best design [127].

All the steps are discussed throughout this thesis. The first step has been elaborated in Chapter 2. The second step is discussed in detail in Chapter 4. The new algorithm design steps are thoroughly elaborated in Chapter 5, in addition to the implementation of the new algorithm.

The fifth step is going to be deliberated in this chapter. This deliberation includes the metrics used in evaluating the performance of the new algorithm, the details of the experiment setup, and the discussion on how the experiments, design, and results are verified and validated. The final step is discussed in Chapter 6.



**Figure 3.2:** Performance evaluation approaches (adapted from [3])

### 3.2 Methods used in Evaluating Network Performance

The most essential issue in network performance evaluation is to choose the suitable measurement approach, representative measurement settings, and a correct implementation technique, as mentioned in [128]. Performance evaluation of a communication network can be conducted using three possible approaches, namely measurement, experimentation, and simulation, as published by [129, 130, 131]. However, Jain in [1] outlined the possible options as analytical modeling, measurement, and simulation. According to Hassan and Kara in [3], the measurement can be done on an actual network or lab-testing using test beds. To simplify this discussion, the preferred categorization are as follows; mathematical modeling, experimentation, and simulation. In addition, the experimentation can be done over an actual network or on a test-bed network. Figure 3.2 illustrates the categorization of typical research evaluation approaches.

The analytical modeling technique, as mentioned in [132], is a set of equations describing the mechanism of a computer network system under study, which is expressed using mathematical symbolism. The steps include building, solving, and validating the analytical model of the mechanism to explore and solve the problem.

This might lead to a better decision of the system before the actual implementation process.

The experimentation or measurement approach, as discussed in [127], measures the performance of a real communication system or a prototype of the communication protocol. According to the author, by using this technique, a researcher can design prototypes of a communication network system and test them in a specific network environment. This can be done in a lab or a test bed (using network emulator) or on an actual network. The performance of the prototype would be monitored and evaluated during and after the running of the prototype. According to Williamson in [133], network traffic measurement provides a means to understand what is or is not working properly in a local-area or wide-area network. By using specialized network measurement hardware or software, a network researcher can collect detailed information about the transmission of packets on the network, including their timing structure and contents. With detailed packet-level measurements, and some knowledge of the Internet protocol stack, it is possible to “reverse engineer” significant information about the structure of an Internet application, or the behavior of an Internet user.

Simulation by definition is an approximation or abstraction of the real world, as mentioned in [134]. According to Lynn [135], simulation is a representation of an item that consists of equipment, devices, systems, or subsystems in a realistic form. Network simulation enables the researchers to perform experiments on the operations of the targeted network model without the possibility of disrupting the actual network. In another way, Al-Momani [132] defined simulation as a discipline of designing a model of theoretical communication network system, executing the model, and analyzing the execution output using network simulation software. Simulation uses a computer-generated system to represent the dynamic responses and behaviours of a real or proposed system.

Besides the above mentioned issues, it is necessary to evaluate the network performance under a controlled condition. This is due to the complexity of today's networks, especially the Internet, and the fact that transport protocols have become more and more complex. With the increase in complexity it is nowadays hard, if possible at all, to fully grasp the behavior and performance issues of a protocol/algorithm, as discussed in [136].

### **3.2.1 Network Performance Evaluation Approach Consideration**

According to Jain in [1], among the three performance evaluation methods (mathematical modeling, measurement and simulation), mathematical modeling requires the least time to perform performance evaluation. Performance evaluation with measurement can take a shorter or longer time depending on the complexities and difficulties of performance evaluation, while the time taken to perform the simulation is moderate.

In terms of accuracy, performance evaluation with mathematical modeling has the lowest accuracy. The accuracy of measurement is varied; it can be very high accuracy, but it also can be of low accuracy. Although the measurement uses real hardware and software settings, it may not give an accurate result. Some parameters used in the system configuration, for example system configuration, type of workload, and time of the measurement may be unique for the experiment, but it may not represent the actual range of variables found in the Internet. Simulation can incorporate a more detailed description of the system than a mathematical model, and it also requires fewer assumptions than the mathematical modeling. The simulation model can be re-modeled until the real implementation is satisfied. This makes it closer to reality.

According to Law [137], mathematical modeling is the preferable evaluation technique if it can provide efficient computation of a communication network system. Thus, it may be suitable for modeling a simple system, as mentioned in [138], but



it is difficult to describe a complex system such as network communication protocols. However, communication network systems are too complex to be realistically modeled and evaluated with the analytical technique [1]. In addition, Keshav in [139] stated that mathematical modeling has two major drawbacks. Firstly, it makes too many simplifications and assumptions, which inhibit the model from representing the actual behavior of the system. The second drawback is it ignores interaction, which is the main factor that affects the communication network system performance. Apparently, this method suitable to be used in order to study simple system and overall characteristics of the system behavior, and if it is used to study a complex system, a great deal of simplifications and assumptions are required [137].

Although performing experiments in a real network are very interesting and more representative than other approaches, it has its own constraints. The following is the major impediments to the employment of measurement technique for communication network system performance evaluation, as discussed in [128, 127, 140, 137, 141]:

- Developing and maintaining a test bed requires a lot of resources, money, and time. Only a small number of researchers, usually financially backed by big telecommunication companies, can afford this evaluation technique.
- Scaling of actual network in a test bed is not possible. This is due to the fact that it involves hundreds of nodes, in addition to a large physical area that is required for setting up a multi-hop topology.
- Experimenting a prototype with a test bed is difficult to reconfigure because of its inflexibility. Moreover, it is also difficult to replicate and share it with other researchers.
- It is too costly to test a new untested network protocol or algorithm in the actual network, if it goes wrong the consequence would be very disruptive to the whole network.

However, Williamson in [133], specified four main reasons where network traffic measurement is considered a useful methodology:

- **Network Troubleshooting:**

Since the real problem is most probably related to the network setup, thus the process of identifying the problem should be conducted in the real network.

- **Protocol Debugging:**

Developers often want to test out “new, improved” versions of network applications or protocols (on a test bed measurement). Network traffic measurement provides a means to ensure the correct operation of the new protocol or application, its conformance to required standards, and (if necessary) its backward-compatibility with previous versions, prior to unleashing it on a production network.

- **Workload Characterization:**

Network traffic measurements can be used as input to the workload characterization process, which analyzes empirical data (often using statistical techniques) to extract salient and representative properties describing a network application or protocol. Knowledge of the workload characteristics can then lead to the design of better protocols or networks for supporting the application.

- **Performance Evaluation:**

Finally, network traffic measurement can be used to determine how well a given protocol or application is performing in the Internet. Detail analysis of network measurements can help identify performance bottlenecks. Once these performance problems are addressed, a new version of a protocol can provide better (i.e., faster) performance for the end users of Internet applications.

By considering the above-mentioned facts and practices of previous similar research (in the domain of video transmission research), simulation is deemed as being able to meet the requirements of communication system modeling and meet the objectives of this research. Simulation enables the construction of a model that accurately represents the complex behavior of a computer network. This will allow efficient investigation of network performance. Simulation of network protocols with numerous combinations of parameters and interactions can be done without consuming much time and resources. The simulation generates raw data that can be analyzed using statistical tools.

In order to simplify the performance evaluation technique options, Jain in [1] has presented the options as listed in Table 3.1.

**Table 3.1:** Criteria for selecting performance evaluation technique (adapted from [1])

Criterion	Analytical Modeling	Simulation	Measurement
Stage	Any	Any	Post-prototype
Time required	Small	Medium	Varies
Tools	Analysts	Computer languages	Instrumentation
Accuracy	Low	Moderate	Varies
Trade-off evaluation	Easy	Moderate	Difficult
Cost	Small	Medium	High

### 3.2.2 Justification for Simulation Method

According to Kelton et al. in [142], computer simulation is a method for studying a wide variety models of real-world systems by using certain simulation software designed to imitate the system's operation of characteristics. Using a network simulator, a model of communication network system can be designed and modified easily. Modification or alteration of the model can be done by remodeling and changing certain parts of the model. The modified model can be re-evaluated until the desired outcomes are achieved. This allows exploration of complex scenarios that ensure

correctness of the analysis and evaluation.

A close network environment, although interesting, it is difficult to develop. This is due to the difficulty in writing the codes to a “real computer” and to make it communicate/compatible with the computer network protocol suite. Moreover, the close network environment is quite rigid in terms of performing various testing scenarios. According to Magoni and Pansiot [143], a simulation framework provides a sandbox where a harmful design flaw can easily be detected and removed. This is done prior to implementation and experimentation in an operational environment as it is easier and cheaper to perform. However, they made a cautious remark that the simulation results can be distorted if the simulation model is unrealistic.

There are a number of simulation techniques available, e.g. Monte-Carlo, trace-driven, and discrete-event simulations, as mentioned in [1]. For computer network models, the discrete-event approach is the most suitable, as discussed in [137, 144]. In the field of communication network system, simulation has emerged as a main research methodology used by many researchers. Many research works using network simulators have been published in credible publications such as ACM SIGCOMM, IEEE INFOCOM, IEEE/ACM Transactions on Networking, and Performance Evaluation Journal. This is evidence that simulation has been accepted as a solid research methodology, as stated in [144].

According to Al-momani in [132], simulation is one of the most widely used techniques for performance evaluation of complex protocols, and it is the basic tool to explore the proposed protocols and mechanisms in environments that have not been implanted in the current Internet. In addition, he cited Lane et al. in [145] that simulation is ranked as one of the three most important research techniques from 1973 to 1988.

The Internet is a very large and complex network system. It is a highly heterogeneous network with wide range of applications, traffic, protocols, and

network settings. The type and amount of traffic traveling on the Internet path can change rapidly in a very short time. To model such a very complex system with mathematical modeling would almost be impossible without compromising many real world parameters. Simulation as a technique to model complex systems can play an important role in characterizing the behavior of the Internet and the possible effect of the proposed changes of any of its components, as explained in [129]. Simulation can be used to explore a new protocol that has not yet been realized in the Internet, in diversified environments or in large-scale topologies, as expressed by [140]. Therefore, a network simulator can be a very beneficial tool to model and evaluate a new protocol.

The real power of simulation lies in its ability to accurately represent the behavior of complex networks and protocols. In the case of video application simulation, it allows for efficient investigation on the network and protocol design, analysis, and optimization. Using the simulation technique, network researchers can freely explore numerous combinations of parameters and analyze action sequences efficiently and quickly. Researchers can also probe the consequences of protocol designs, carry out analysis of their performances and improve the designs of complex systems.

According to Floyd and Paxson in [129], due to the network's complexity, simulation plays a vital role in attempting to characterize both the behavior of the current Internet and the possible effects of proposed changes to its operation. Furthermore, there is growing concern on having a common evaluation method or using existing measurements and methodologies for Internet research. The reasons are that they will produce a better result comparison, be easier to verify the finding, provide a better understanding of the statistical nature of Internet traffic, and avoid known research pitfalls [146, 141].

There is also a sort of agreement that the Internet is complex, rapidly changing, and heterogeneous, thus the studies on these kinds of environments would be very challenging. It needs carefully-planned and thoroughly understood parameters

involved in order to produce meaningful results, as proposed in [129, 146, 147]. Furthermore, Floyd in [146] said that researchers actually know very little about where Internet congestion occurs, or where it can be expected to occur in the future. There is a need to know better the typical levels of congestion, packet reordering, or packet corruption. All these, demand researchers to perform an Internet evaluation in a more systematic and strategic manner.

Furthermore, as suggested by Floyd et al. in [129], network communication protocol should use a common research method, that is simulation. By using simulation, there are many modules (sub-program to emulate protocols) that have been developed, used, and tested on various research. In addition, the network simulation evaluation technique has benefits over other evaluation techniques for network research, such as improving validation of present protocols and mechanisms, providing plentiful protocol development, and making it easy to study large-scale protocol behaviors. Therefore, in this thesis the simulation technique has been employed to evaluate the video rate control in a communication network system.

In addition to all above-mentioned justifications, the majority of video rate control research is performed using simulation, as mentioned in [37]. The following are some of the works in this area, which employ simulation as the key performance evaluation method [148, 149, 150, 28, 151, 152, 153, 154, 155, 156, 157, 158, 159].

### **3.2.3 Network Simulation 2 (ns2)**

As discussed previously, network simulation has emerged as a main research methodology by many researchers [149, 160, 161, 138, 162]. Their preferences for this method may arise from the real power of simulation in its ability to accurately represent the behavior of complex networks and protocols. So that many researchers are opting for simulation as their research approach for communication network and their findings have been documented in many well known publications [144].

There is a lot of network simulator software available for the purpose of investigating the performance of communication networks. Freely available software includes Realistic And Large (REAL), OMNET++, Global Mobile Simulator (GloMoSim), and Network Simulator 2 (ns2). According to Ke et al. in [37], ns2 is a tool commonly used in network or telecommunication related research communities, where it is an object-oriented, discrete-, and event-driven network simulator developed at the University of California, Berkeley, and written in C++ and OTcl. It covers a very large number of applications, protocols, network types, network elements, and traffic models.

Among the reason ns2 was chosen as a network simulation tool in this study is because ns2 gives the user the ability to easily extend and modified code, it also enables the user to patch in the new developed protocol. After modifying the code, the users then run simulation experiments to verify the efficiency of the new suggested modified protocol. The second reason is because ns2 is open-source software, and it can be freely downloaded from the ns2 website. Besides that, there are many ns2 users with whom the researcher can have discussions to exchange ideas to enhance the research. Moreover, there are many researchers from many counties who have used ns2 as the network simulation tool. In addition, many journals, articles, and papers have been published on ns2 which shows the popularity of ns2 as a simulation tool.

ns2 is a discrete event simulator targeted at network research. It provides substantial support for simulation of the transport protocol, routing, and multicast protocols over wired and wireless (local and satellite) networks. ns2 does not give priority to the Graphical User Interface (GUI) and it is almost a fully text-based tool. Thus, users should write the code by using a text editor. However, since ns2 is free software, it has been used widely by many researchers and its debugging can be easily done.

ns2 simulation results can be observed through graphs for analysis or animations with the Network Animator (NAM) module. NAM is a network animator that was developed for educational purposes and it can display the detailed procedure of the simulation. ns2 has a one-to-one mapping structure between C++ and OTcl. The experiment setting with the ns-2 simulation can be referred in Section 3.4.

### **3.3 Evaluation Metrics**

According to Al-Momani and Ghazali [132, 127], selecting the performance evaluation metrics is a key step and an important part in all performance evaluations. As discussed in Chapter 2, many ways have been adopted by video transmission researchers. Firstly, in terms of either using conventional network-based metrics or video quality related, this work has opted for video quality related metrics. This is due to the fact that video quality related metrics is more direct in evaluating the video performance result.

From the users' perspective, they are more concern about how they experience the video viewing, rather than on how the video effects or utilizes the network. For instance, corrupted bits in the I-frame gives more impact than higher bits corrupted in the B-frame. Although more bits corrupted are revealed in the network-based performance metric, it contributes a lesser impact on the perceived video viewing.

In terms of either to employ objective or subjective video performance evaluation, this work has opted for objective evaluation, as used by many other researchers, particularly in using PSNR [163, 164, 165, 166, 167]. The other metrics which are used many times at many stages of this research are video sequence video rate and QP value. The details on the metrics mentioned will be discussed in the next subsections.



### **3.3.1 PSNR**

As discussed in Chapter 2, this is the most popular metric used in video performance evaluation. PSNR has been chosen as the main metric for the overall video performance in this study because it can be considered as an objective way to evaluate the perceived video quality.

The larger the PSNR value, the better the video quality perceived by the end user. The PSNR value generation is done by using “psnr.exe”. This tool is provided in the Evalvid-RA package, which is available via [168]. This tool calculates the PSNR value for every frame. In Evalvid-RA, the inputs are two video sequences which are the original video and the video sequence after transmission. The PSNR output is a two-column format, which consists of frame number and the PSNR value gained. The average PSNR value is displayed at the end of the output. Figure 3.3 shows an output sample from the tool.

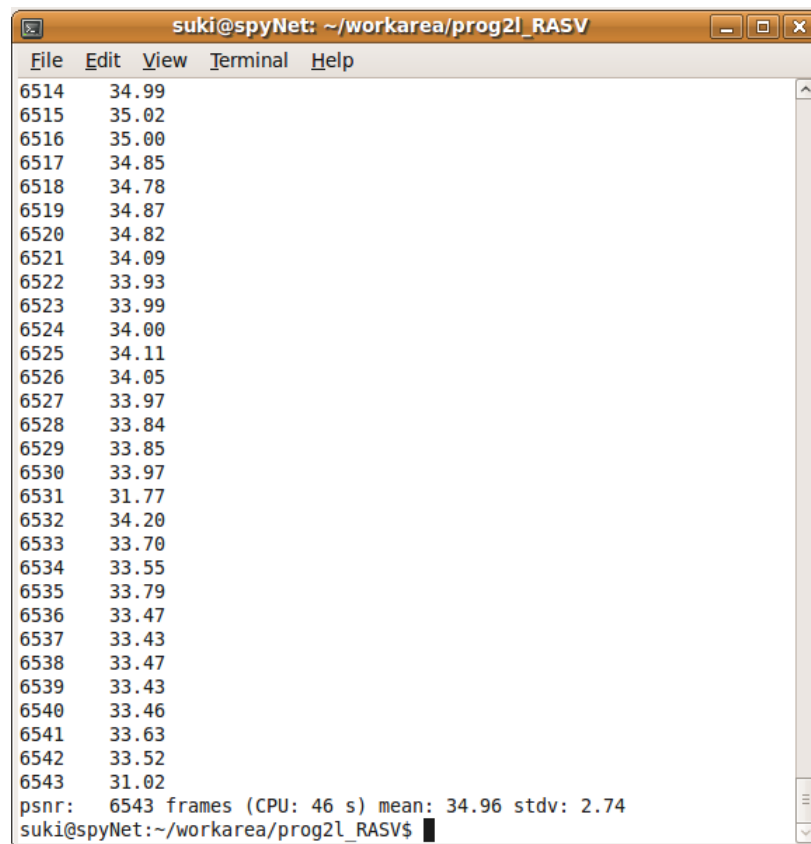
### **3.3.2 Video Rate**

A higher video rate indicates more bits allocated to a video frame. More bits in a video frame means a more detailed, clearer and crispier image in the frame. This definitely contributes to a higher quality of video sequence.

However, a higher video rate should not exceed the overflow level, which may lead to video frame loss. This is more damaging in terms of video quality.

### **3.3.3 QP Stability**

The QP value metric can be used in two ways. Firstly, the lower the QP value, the higher the video rate produced. This is almost a similar metric to the higher video rate metric. Thus, this way of evaluating the video performance is not used often. The second way of QP value metric is used is by examining its stability. The most stable QP



The image shows a terminal window titled "suki@spyNet: ~/workarea/prog2l\_RASV". The window contains a list of frames and their corresponding PSNR values. The frames are numbered from 6514 to 6543. The PSNR values range from 31.02 to 35.00. At the bottom of the list, a summary line shows the total number of frames (6543), the CPU time (46 s), the mean PSNR (34.96), and the standard deviation (2.74). The terminal prompt is "suki@spyNet:~/workarea/prog2l\_RASV\$".

Frame	PSNR
6514	34.99
6515	35.02
6516	35.00
6517	34.85
6518	34.78
6519	34.87
6520	34.82
6521	34.09
6522	33.93
6523	33.99
6524	34.00
6525	34.11
6526	34.05
6527	33.97
6528	33.84
6529	33.85
6530	33.97
6531	31.77
6532	34.20
6533	33.70
6534	33.55
6535	33.79
6536	33.47
6537	33.43
6538	33.47
6539	33.43
6540	33.46
6541	33.63
6542	33.52
6543	31.02

psnr: 6543 frames (CPU: 46 s) mean: 34.96 stdv: 2.74  
suki@spyNet:~/workarea/prog2l\_RASV\$

**Figure 3.3:** Sample output of “psnr.exe”

value is a VBR type of video sequence, where its QP value is the same from beginning to end of the video frames.

The stable QP value provides a smoother video viewing for the users. A consistent QP value provides a same proportion of bits for each frame. Thus, it creates a smooth-consistent user viewing. Otherwise, the user will see different video frame quality.

### **3.4 Experimental Setting**

Since this research focuses on an application layer problem, which is the video transmission application, the experimental setting should be discussed from the top to bottom of TCP/IP layers, in particular, the video and the network setting. In terms of the simulation settings, the setting should specify the video data, video traffic type, and the Internet setting, which includes transport protocol, topology, transmission rate, delay, routing, and etc.

On the video setting part, two main issues are of interest, which are the video sequence and the type of video traffic to be used. The interest in video sequence is what type of video, how long the sequence (the video duration), the video format, and etc. Whereas, the type of video traffic is concerned with either the use of real video data traffic or syntentic data.

#### **3.4.1 Video Traffic**

The video traffic for video transmission research is generally categorized into three different characterized encoded video, namely, as stated in [169, 37, 44], they are called;

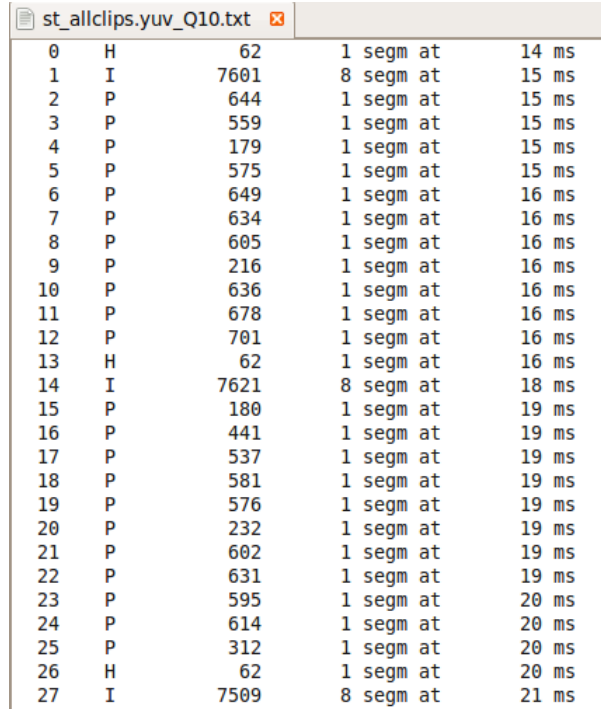
- i. Video Bit Stream,
- ii. Video Traffic Trace, and

### iii. Video Traffic Model.

The Video Bit Stream, which is generated using the encoding mechanisms, contains complete video information including the video bits. It means that this type of traffic uses real video data, which is after the video is encoded, the video is transported as it is. Although it is very interesting to use real video as a video traffic in experiments, like using the whole population of the interest as respondents in social research sampling, in many of the video research settings, they are not practical and resource intensive. As pointed by Seeling et al. in [44], among the problems of employing Video Bit Stream traffic is that it is huge in size. It might be in the size of several GBytes for one hour of compressed video or several tens of GBytes for one hour of uncompressed video. Another problem of the Video Bit Streams is that it is usually proprietary and protected by copyright. This limits the access to other video application researchers, and also hinders the sharing of video data for research experiments among research groups.

It is impractical to transmit the huge video data back to the sender, which can be really large in size. Consequently, Video Traffic Trace can be regarded as an effective alternative to the bit streams, in which they represent the true traffic and quality of the videos. It is an abstraction of the real-video stream, as mentioned in [37]. Therefore, despite using the actual video bits in transporting the video information, the traces only give the number of bits and other information of the individual video frames, such as video frame type and offset time. Thus, there is no copyright issues. The Video Traffic Trace is a very convenient way for video characterization in networking research, as described in [169, 44]. Indeed, they claimed that Video Traffic Trace had stimulated the video networking research. The networking research community experienced an initial explosion in research on video transport after the MPEG-1 traces became publicly available around 1995.

A Video Traffic Trace comprises rows of information, where each row typically contains of the frame number, the frame type (I, P, or B), the frame size usually in bytes, and the time offset of the frame. Figure 3.4 displays a typical video trace.



Frame Number	Frame Type	Frame Size (bytes)	Time Offset (ms)
0	H	62	14 ms
1	I	7601	15 ms
2	P	644	15 ms
3	P	559	15 ms
4	P	179	15 ms
5	P	575	15 ms
6	P	649	16 ms
7	P	634	16 ms
8	P	605	16 ms
9	P	216	16 ms
10	P	636	16 ms
11	P	678	16 ms
12	P	701	16 ms
13	H	62	16 ms
14	I	7621	18 ms
15	P	180	19 ms
16	P	441	19 ms
17	P	537	19 ms
18	P	581	19 ms
19	P	576	19 ms
20	P	232	19 ms
21	P	602	19 ms
22	P	631	19 ms
23	P	595	20 ms
24	P	614	20 ms
25	P	312	20 ms
26	H	62	20 ms
27	I	7509	21 ms

**Figure 3.4:** A typical video trace

On the other hand, a Video Traffic Model is typically developed based on the statistical properties of a set of video trace samples of real video traffic, as written in [4]. The developed Video Traffic Model is verified by comparing the traffic it generates with the video traces. If the Video Traffic Model is deemed sufficiently accurate, it can be used for mathematical analysis of networks, for model driven simulations, and also for generating so-called virtual (synthetic) video traces. Transform Expand Sample (TES) is an example of this kind of traffic for generating data that closely match (in terms of marginal distribution and autocorrelation function). The developed model can be used for mathematical analysis of networks, but it lacks the possibility of visualizing a transmitted video, as stated in [37].

The video traffic used in this research is the Video Traffic Trace. The idea is to employ real video information (from the selected video sequence) but without the use of very big data in the experiments. Compared to traffic model, trace-traffic is considered credible as it represents an actual traffic load, as justified in [170]. The video traffic trace perfectly fixes the requirement and it has been employed by the majority of video communication research community, as in [163, 171, 123, 167].

### **3.4.2 Video Sequences Used**

The video sequences used are a combination of several video clips taken from <http://trace.eas.asu.edu/yuv/index.html>. These include “news”, “bridge\_far”, “bridge\_close”, “bus”, and “highway”. They are typical video sequences used in various video rate control studies. All the video clips and its profiles are describe in Table 3.2. The Motion categories profile of the video sequence is based on motion activity in the sequence. The descriptions of the activity are mentioned in the Description column.

Five frames are taken from each sequence to display the snapshots of the clips. The snapshot of the sequences’ frames can be reffered in Appendix A. The combined clip duration is 218.4 seconds, which consists of 7099 frames (6552 frames without header) and 547 GoPs. All clips used are in a raw YUV video format, 25 frames per seconds (fps) and in CIF format (352x288). CIF and QCIF format are two commonly used formats in video transmission-related studies [37, 44].

### **3.4.3 Network Simulation Setup**

The video data will be sent over the network until all the video sequences are completed. It might effect the data loss and the transmission duration. Therefore, the simulation has to take into account the network aspect and should simulate the real network as close as possible. From the aspect of the network simulation setup

**Table 3.2:** Profile of video sequences used in the experiments

Video clips	Size (frames)	Motion categories	Description
News	300	Medium	Two news reporter are talking.
Bridge (far)	2101	Low	A bridge view from far side, then a small boat going to that direction with a bird flying in the sky.
Bridge (close)	2001	Low	A bridge view from near side and many people are walking across the bridge.
Bus	150	Very High	A bus moving speedily in a road.
Highway	2000	High	A panoramic view from a vehicle speedily moving in a highway.

regarding the scope of the study in video rate shaping, all the network parameters have been setup to closely imitate the real Internet, wherever possible and related.

The discussion in this subsection will include the transmission protocol that has been employed, the network topology, and all the related parameter settings.

#### **3.4.3.1 The Transmission Protocol**

The transmission (or transport) protocol and its congestion control that has been deployed for this study is TCP-Friendly Rate Control (TFRC). This protocol has been defined and refined in RFC 5348, “TCP Friendly Rate Control (TFRC): Protocol Specification” (refer to [103]). As stated in [172, 173], TFRC is a congestion control mechanism designed for unicast flows operating in an Internet environment and competing with other TCP traffic.

With the advent of TFRC, both advantages for video application of TCP and UDP are gained. As mentioned in [103], TFRC is designed to be reasonably fair when competing for bandwidth with TCP flows. However, TFRC has a much lower variation of throughput over time compared with TCP, which makes it more suitable for a real-time application, such as video transmission, where a relatively smooth sending rate is of importance. Moreover, as stated further in [103], TFRC is designed for best

performance with applications that use a fixed segment size, and vary their sending rate in packets per second in response to congestion. TFRC can also be used with applications that do not have a fixed segment size, but where the segment size varies according to the needs of the application (e.g. video applications).

Therefore, for this study (video transmission application) which requires lower variation of throughput, varied segments of data, unicasting, and TCP-friendly protocol, TFRC can be regarded as one of the best options. Furthermore, many other researchers in the field of video transmission application also adopted TFRC as their transmission protocol, such as [174, 175, 31, 32, 176, 177, 178, 179].

#### **3.4.3.2 Network Simulation Topology**

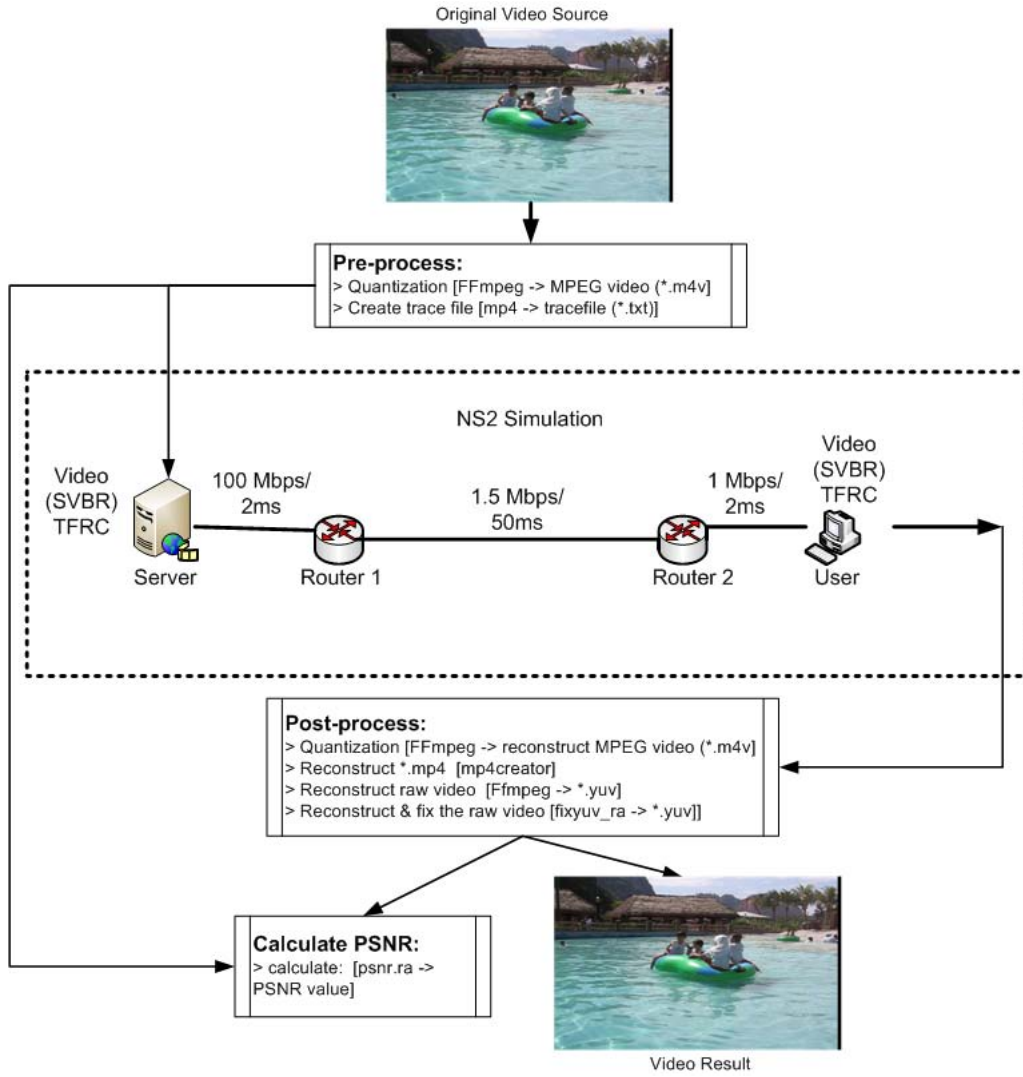
In simulation, a network is typically illustrated as an undirected graph. The topology of the graph is a description of the way the nodes are connected together. Properties, such as the average node degree and the diameter, give information on a graph topology. For network protocols, the knowledge of the topology of the medium is very important as it directly translates into useful information such as path bandwidth and path delay.

Choosing an appropriate network topology in simulating communication network systems is very important. The correct network topology ensures that it represents the problems under investigation, and the simulation results are as general as possible, as discussed in [127, 180, 143].

In the case of the new video rate shaping algorithm, the well-known single bottleneck dumb-bell topology has been employed. This topology is then applied in Evalvid-RA environment. Figure 3.5 demonstrates the network setup for the experimental purpose.

This topology is believed sufficient for this study because the video rate shaping algorithm does not getting any feedback from the network, as explained in the scope of this study. Moreover, the dumb-bell topology was also used by other video





**Figure 3.5:** Simulation parameter setting

transmission researchers in their study [181, 69, 118, 182, 183, 184, 177, 185, 186].

### 3.4.3.3 Network Simulation Settings

As mentioned in [143], the best way to draw acceptable conclusions would be to use a topology that is closest to the real network where the new protocol will be deployed. Therefore, the network settings in the simulation topology should closely match the real Internet environment from the aspect of parameter settings which have been assigned in the experimental setup.

The propagation delays of the access link is 2 ms while the delay of the link between the gateways, which is one-way delay, is 50 ms. Thus, the end-to-end round trip propagation delay will be 104 ms. This value closely represents the typical WAN delay on the Internet, which is a 105 ms propagation delay [187].

For the bandwidth of the links, 1 Mbps is used on the link between the user and its gateway to represent the typical broadband home Internet access speed. The capacity of the link between the server and its gateway is 100 Mbps which represents a typical LAN speed. The capacity of the link between the gateways is 1.5 Mbps; this is to represent T1 transmission rate. This bandwidth is sufficiently provisioned so that congestion only occurs on the link between routers, as suggested in [33]. All these parameter settings are shown in Figure 3.5.

Regarding the queuing policy, the queue size of the gateways is set to a value which is twice the Bandwidth Delay Product (BDP) of the connection, as suggested by Byers et al. in [188] and used in [189, 33]. Therefore, the queue size of router 1 is 74.4 packets and 49.6 packets for the router 2 queue size.

### **3.5 Validation and Verification**

IEEE standard Board, as stated in [190], has defined validation and verification as a process of verifying the internal consistency and correctness of data, and validating that it represents real-world entities appropriate for its intended purpose or expected range of purposes. Whereas within the modeling and simulation community, as written in [134], validation is the process of determining the degree to which a model, simulation, or federation of models and simulations, and their associated data accurately represent the real world from the perspective of its intended use(s). Verification is the process of determining that a computer model, simulation, or federation of model and simulation implementations and their associated data accurately represent the

developer's conceptual description and specifications.

According to Ghazali in [127], validation is a process that the conceptual model accurately represents the system under study, in order to provide meaningful answers for the questions being investigated. Whereas in [191], validation is a determination of whether the conceptual model can be substituted for the real system for the purpose of experimentation. It is the first step to ensure that the assumptions are realistic. As written by many authors, including Banks in [191], verification is a determination of whether a conceptual model is correct. Thus, it is a step to ensure that the model implements the assumptions correctly. This involves examination of the simulation program to ensure that the operational model accurately reflects the conceptual model.

### **3.5.1 Validation of ns-2**

As highlighted by many network researchers, such as [132, 128, 127], ns-2 is packaged with a large collection of detailed validation scripts that can be used by researchers to validate the ns-2 after the building process of ns-2. During the validation process, the simulator is run using a specific set of input values with known output values. Then, the results obtained from the simulation are compared with the known output. If the results match the known output, then the ns-2 is valid. Nonetheless, if it fails or some of the comparisons do not match, the users will be notified and asked to repeat the validation process for the failed models. This process is repeated until the validation process is passed successfully. The ns-2 used in this research has been validated using this validation process.

### **3.5.2 Validation of the Simulation**

As stated in [134], simulations are an approximation or abstraction of the real world, and simulation validation in particular is the process of determining the degree to which a model or simulation is an accurate representation of the real-world. As discussed in

the previous section, that the simulation settings in this study are using real-world parameters. In terms of application layer data, it also uses real-world video sequences, which have been used by many other researchers, albeit the traffic is in the form of a video trace.

### **3.5.3 The New Algorithm Implementation Verification and Validation**

In designing the new algorithm, which is discussed comprehensively in Chapter 5, the design objectives are stated. Each objective has its own sub-algorithm, principle, or formulation. For each sub-algorithm, the verification is done by evaluating the result to observe whether it conforms to the initial objective or formulation. All the sub-algorithms have been verified and the details of the verification process are discussed in Chapter 5. Furthermore, the new algorithm implementation is validated on ns-2 by using Run-time Trace and Incremental Implementation. For every simulation, the Run-time Trace is checked to ensure it runs as expected.

The validation is done by using multiple types of video sequences. The video test sequence, as discussed previously, is a combination of several video clips. As discussed at length in Chapter 6, the result of the whole sub-algorithms of the new algorithm have performed as they are intended, thus it can be concluded that the new algorithm is validated.

## **3.6 Summary**

This chapter has described the approach that has been used in ensuring that the research objectives can be fulfilled, verified, and validated. This chapter started with six research steps (in achieving the research objectives), which are; perform an in-depth study on the issues surrounding video rate control, perform an empirical analysis on the previous video rate shaping algorithm (SVBR), design a new shaped control algorithm,

implement, validate, and verify the design of the new shaped algorithm, execute the experiments, and lastly carry out performance evaluation of the new algorithm by comparing it with the SVBR algorithm.

In the evaluation of the new algorithm, this study has opted for the simulation approach, which had been employed by many other researchers in this area. In terms of evaluation metrics, this study chose to use video related metrics, rather than network-based metrics. The video related metrics that have been selected are PSNR, higher video rate, QP stability, and frame loss.

In performing the experiments by using the simulation approach, great effort has been taken to ensure that the simulation is representing the real world. The real video data is used, although the traffic is in the form of a video trace. The network setting also use the parameters that closely imitate the real Internet “variables and values”.

This chapter has also explained how the results are verified and validated. All the algorithms have been inspected so that they meet the intended or the original requirements of the designed objectives. The video sequences comprise several clips with different activities. The ns2 package plus the new implementation is validated using the built in ns2 validator.

The next chapter discusses comprehensively the empirical analysis of the SVBR algorithm. By understanding thoroughly its strengths and weaknesses, the new shaped algorithm for the video application can be effectively designed.

# **CHAPTER FOUR**

## **EMPIRICAL EVALUATION OF THE SHAPED VARIABLE BIT RATE (SVBR) ALGORITHM**

After describing the research methodology for performance evaluation of the new Shaped Variable Bit Rate (SVBR), extensive analysis and evaluation of the Shaped VBR algorithm is presented in this chapter. As mentioned in the Chapter 2, many recent works, which used the original SVBR, made no changes to the core of the algorithm itself. This extensive analysis reveals the strengths and the weaknesses of the algorithm. Consequently, this study leads to a set of potential enhancement to the original SVBR algorithm, that can be applied in a slight delay video transmission application.

The analysis starts by underlining the background of the SVBR. It then describes the SVBR algorithm, in terms of its principles, how it determines a suitable bit rate, and quantization parameter (QP) value. In addition, the chapter presents the experiments conducted to compare the SVBR performance with traditional rate controllers, namely Variable Bit Rate (VBR) and Constant Bit Rate (CBR). Finally, the SVBR algorithm is empirically analyzed to identify its strengths and weaknesses. From the analysis, the important areas which can be enhanced are highlighted.

## 4.1 Exploiting the Advantages of CBR and VBR

As stated in the previous chapter, most of the video applications employ video rate controllers in the form of either a VBR or CBR. The advantage of VBR is that it produces a consistent visual quality, while CBR generates constant video rate for the network interface as its main advantage.

However, the bursty form of VBR causes grave problems to networks in terms of significant variation of the network traffic, jitters and delays. Equally, the problems with a CBR implementation are the additional delay due to buffering, and the visual quality tends to vary according to video content.

There is an obvious need for an alternative solution by taking advantage of both CBR and VBR. At the same time, the alternative solution should eliminate the weaknesses of both rate controllers. Therefore, an ideal rate controller requires a higher and consistent visual quality video, video rate is always within a permissible bandwidth level, and there is less delay. Delay can happen either because of the introduction of an additional buffer or complex computational algorithm.

Therefore, it is useful to try to encode the video with an open loop VBR as much as possible, but, at the same time, there is a need to control traffic admission into the network when the permissible level is exceeded. By doing so, it helps to maintain the consistent visual quality in VBR and gets into reasonable compromise with adaptive quality when the network transmission rate degrades. In addition, it avoids unpredictable large bursty rate variations, as in VBR. However, it is done without the rigidity and systematic coding delay of CBR coders or intermediate CBR buffer.

SVBR was introduced by Hamdi et al. [23] in 1997. The main idea behind SVBR is to limit the open-loop burst while, at the same time, allowing open-loop VBR coding, provided that they are still within a permitted constraint. To achieve this objective, SVBR manipulates a leaky bucket algorithm to perform admission control.

The leaky bucket used in SVBR can be considered as an imaginary buffer, thus, no extra delay is introduced. Moreover, Hamdi et al. assumed that for a fast moving scene with complex image structure, the scene quality can slightly be reduced, since human eyes do not have enough time to notice the image details. In addition, Hamdi et al. had suggested applying the algorithm at Group of Picture (GoP) granularity, which consequently yields a less complex algorithm and lower delay.

## 4.2 SVBR Algorithm

In order to evaluate the SVBR algorithm performance, this subsection presents the primary foundation of the SVBR algorithm. These include its principles, its mechanics of determining video rate, and the complete steps of the algorithm.

### 4.2.1 SVBR Principles

It can be concluded that SVBR principles revolve around several principles. Firstly, the algorithm works on GoP granularity, thus, it has low complexity that leads to a lesser delay [2]. Therefore, it is much simpler than the one employed in the CBR coder which works on macroblock-by-macroblock variations [23].

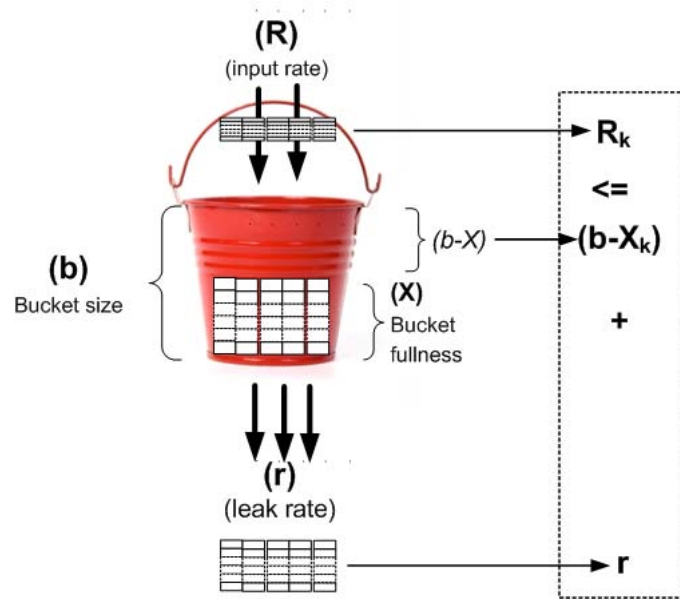
#### 4.2.1.1 Controlling Video Data Input

The second principle of SVBR algorithm is that it uses a leaky-bucket algorithm to avoid an excessive burst video rate into the network. By using the leaky-bucket algorithm, the SVBR mechanism can control the video data admission into the network. Thus, it avoids data loss at the network interface. This principle can be written as follows,

$$R_k \leq r_k + (b - X_k) \quad (4.1)$$



where  $R_k$  is video data input,  $r_k$  is the leak rate,  $b$  is the bucket size and  $X_k$  is the level of bucket fullness after transmitting the data at the time of  $k$ . The relationship of all variables are illustrated in Figure 4.1. The  $(b - X_k)$  in Equation 4.1 is the space in the bucket that can still accommodate more video data. Whereas,  $r_k$  is the data from the bucket that has been sent into the network interface. Thus, the blank space in the bucket is now  $(b - X_k) + r_k$ . Therefore, Equation 4.1 restricts the video data input, so that the bucket fullness will not be overflowed. Consequently, no drops will occur.



**Figure 4.1:** Restrict input in the leaky-bucket algorithm

In order to realize the above-mentioned principle, the video bit allocation or  $R_k$  should be controlled so that the  $X_k$  does not exceed  $b$ . For that purpose, Hamdi et al. in [23] propose the following equation,

$$X(k+1) = \min \{b, (\max \{0, X(k) - r\} + R(k))\} \quad (4.2)$$

Here, they defined  $X(k)$  as a bucket fullness level at the beginning of GoP- $k^{th}$  video data transmission (before transmitting GoP- $k^{th}$  data). Thus,  $X(k+1)$  can be regarded as the bucket fullness level after transmitting GoP- $k^{th}$  data. Equation 4.2 restricts the

bucket fullness level into Equation 4.3.

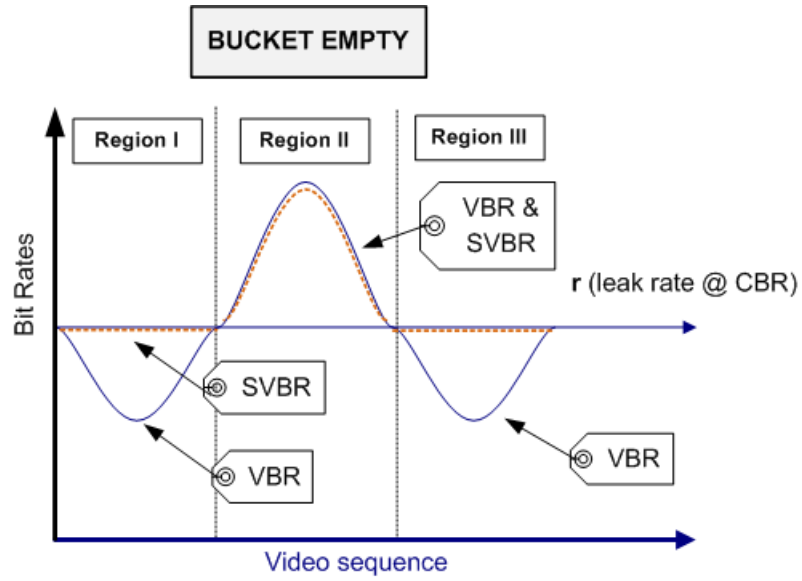
$$0 \leq X(k+1) \leq b \quad (4.3)$$

Here Equation 4.3 can be realized since the mathematical expression  $\max\{0, X(k) - r\}$  from Equation 4.2 will produce a positive result between 0 and  $(X(k) - r)$ , even if the expression  $(X(k) - r)$  produces a negative result. Even though expression  $(\max\{0, X(k) - r\} + R(k))$  might produce a value which is higher than  $b$ , the expression  $\min\{b, (\max\{0, X(k) - r\} + R(k))\}$  will cap the result to  $b$  only.

The important thing here is to determine the input rate  $R(k)$ , which determines the bucket fullness level. The bucket fullness level needs to be controlled so that it is neither too close to  $b$  nor too close to zero. This is because video rate changes most drastically as the leaky-bucket level approaches its limits, i.e. zero or  $b$ . In the former case, the rate controller tends to be at CBR rate controller, which is at the rate  $r$ . Whereas, when the bucket fullness level is approaching zero, the coder does not fully use the available average bit rate. Therefore, in [23], Hamdi et al. proposed that the adjustments to  $R(k)$  should allow flexible, open-loop-like control when  $X(k)$  is in a middle range around  $b/2$  while attracting it back to this range if it tends to approach either extremes, i.e. zero or  $b$ .

#### 4.2.1.2 Allowing VBR Coding when the Network Permits

This principle can be written as follows: a video sequence with reasonable activity and duration can be coded at the normal VBR rate while excessively long and/or active sequence is “truncated” and their bit rate is reduced to  $r$ . This means that for a video sequence that conforms to the traffic contract, the shaping algorithm behaves like a normal VBR. On the other hand, during overload periods (i.e. video sequence does not conform to the traffic contract), the algorithm aims to bring the rate down to  $r$ . However, because network resources are dimensioned based on the leaky-bucket conformance, this shaping avoids data loss. Thus, only harmful sequences are shaped.

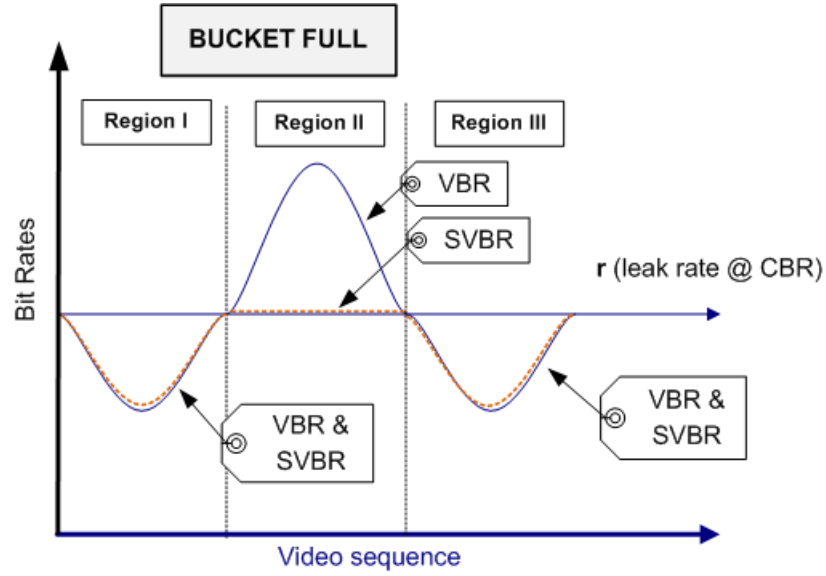


**Figure 4.2:** SVBR shaping principle when bucket empty

This principle can be depicted in Figure 4.2 and Figure 4.3. SVBR video rates are shown in the red-dotted lines. Here, SVBR uses bucket fullness level as a conditional parameter to address this principle. When the bucket is empty, it indicates a video sequence with a reasonable activity. Whereas, the full bucket shows that active video sequence is being processed.

This principle can be easily seen in the Region II of Figure 4.2. When the bucket is empty, SVBR uses the VBR video rate. Thus, the higher quality of VBR in Region II is maintained. On the contrary, the video rate is decreased into the CBR rate when an excessive active sequence has occurred, as illustrated in Region II of Figure 4.3. Here, SVBR compromises high visual quality since the bucket is already full.

In the case of low VBR video rate (lower than CBR rate), as seen in Region I and Region III of Figure 4.3, SVBR applies the normal VBR video rate. This will not make the bucket overflow, since the leak rate is greater than the input rate. However, in a similar case but with an empty bucket level, SVBR employs CBR rate. This will increase the video rate which leads to a better visual quality without the risk of bucket overflow or underflow.



**Figure 4.3:** SVBR shaping principle when bucket full

However, in the actual implementation, the bucket is rarely completely empty or full. Thus, by using linear calculation, which will be described later, the SVBR will be lying between the CBR and VBR video rates. This implementation has advantages and disadvantages, which is going to be analyzed further in this chapter.

#### 4.2.2 Determining Suitable Bit Rate Allocation and Quantization Parameter Values

As discussed in Chapter 2, two main aspects in video rate control research are bit rate allocation and QP [13, 14, 19]. SVBR uses estimation and prediction in calculating the next GoP QP value and video bit rate allocation, which means determining  $R(k+1)$ . The size of  $R(k)$  is very much related to QP.

It should be noted here that the term “bit rate allocation” is a general or typical term, used to describe the video rate. However, it should be stressed here that the video rate used in the implementation of the SVBR algorithm is Bytes/Gop.

The notation  $Q(k)$  is used to represent the quantization parameter value used in the  $k^{th}$  GoP.  $R_{est}(k+1)$  notation will be used to represent the estimation video rate as compared to  $R(k)$  as actual video rate.

#### 4.2.2.1 Determining Bit Rate Allocation

As described in Subsection 4.2.1.2, the SVBR video rate lies between CBR and VBR video rates due to the use of linear calculation. Based on the bucket fullness level, SVBR video rate can be either close to CBR or VBR. For that purpose, SVBR uses Equation 4.4 and Equation 4.5.

if  $R_{open}(k+1) > r$

$$R_{est}(k+1) = (1 - \varepsilon(x)) \cdot R_{open}(k+1) + \varepsilon(x) \cdot r \quad (4.4)$$

if  $R_{open}(k+1) \leq r$

$$R_{est}(k+1) = \varepsilon(x) \cdot R_{open}(k+1) + (1 - \varepsilon(x)) \cdot r \quad (4.5)$$

Here  $\varepsilon(x)$  is a simple function to calculate the ratio of bucket fullness level. The calculation is shown in Equation 4.6:

$$\varepsilon(x) = \frac{X(k+1)}{b} \quad (4.6)$$

Whereas,  $R_{open}(k+1)$  is actually a GoP- $(k+1)^{th}$  VBR video rate estimation. Since SVBR is targeted for real time video application, where the next GoP data is not known in advance. Thus,  $R_{open}(k+1)$  is needed in order to calculate  $R(k+1)$ . For that purpose, SVBR uses Equation 4.7 as a prediction or estimation for the  $R_{open}(k+1)$ .

$$R_{open}(k+1) = \frac{R(k) \cdot Q(k)}{q} \quad (4.7)$$

This equation is formulated as above in order to indicate that SVBR will use quantization parameter  $q$  for the generation of VBR video rate as its base rate (VBR uses a constant QP or  $q$  for the whole video sequence).  $q$  is any suitable value that will be fixed by the user in the beginning of the operation. The details of the algorithm can be referred in [23, 7].

As described previously, the purpose of the bucket fullness function is to determine

whether it should be closer to CBR ( $r$ ) or VBR ( $R_{open}(k+1)$ ). If the bucket fullness is high ( $\varepsilon(x) > 0.5$ ) and  $R_{open}(k+1) > r$  (refer to Region II of Figure 4.3), the SVBR video rate should be closer to  $r$ . This is reflected well in Equation 4.4; higher  $\varepsilon(x)$  value increases  $r$  value and comparatively will decrease  $R_{open}(k+1)$  value by using expression  $(1 - \varepsilon(x))$ . On the other hand, if the bucket fullness is low ( $\varepsilon(x) < 0.5$ ) and  $R_{open}(k+1) \leq r$  (refer to Region I and III of Figure 4.3), the SVBR video rate is closer to  $R_{open}(k+1)$ . This condition is reflected in Equation 4.4. In all combinations of bucket fullness levels,  $R_{open}(k+1)$  is greater or lower than  $r$ , and either SVBR is closer to  $r$  or  $R_{open}(k+1)$ , as shown in Table 4.1.

**Table 4.1:** Relationship of bucket fullness, active sequence, and SVBR video rate

	$\varepsilon(x) < 0.5$	$\varepsilon(x) > 0.5$
$R_{open}(k+1) > r$	SVBR is closer to $R_{open}$	SVBR is closer to $r$
$R_{open}(k+1) \leq r$	SVBR is closer to $r$	SVBR is closer to $R_{open}$

From Equation 4.7, it is clear that the next GoP  $R_{open}$  is calculated based on previous  $R(k)$  and QP values.

#### 4.2.2.2 Determining QP Value

In determining the next QP value, which is vital in producing the suitable bit rate allocation for the next GoP data, the following equations are used;

if  $R_{open}(k+1) > r$

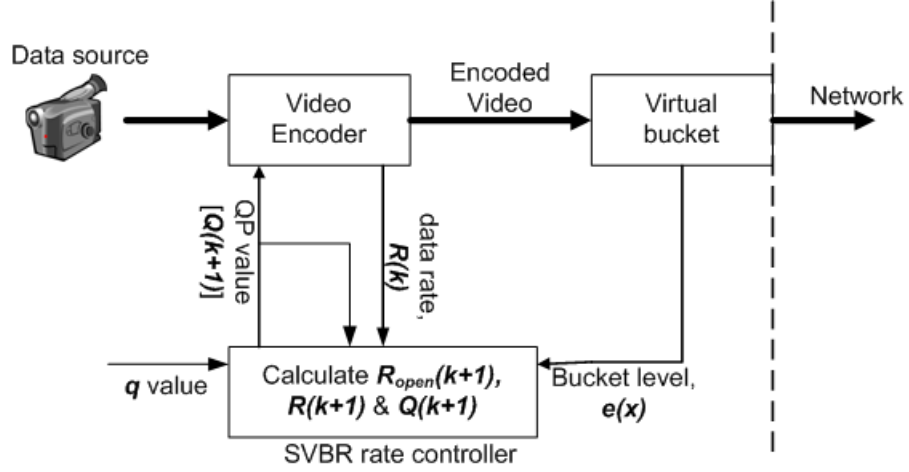
$$Q(k+1) = \frac{q \cdot R_{open}(k+1)}{(1 - \varepsilon(x)) \cdot R_{open}(k+1) + \varepsilon(x) \cdot r} \quad (4.8)$$

if  $R_{open}(k+1) \leq r$

$$Q(k+1) = \frac{q \cdot R_{open}(k+1)}{\varepsilon(x) \cdot R_{open}(k+1) + (1 - \varepsilon(x)) \cdot r} \quad (4.9)$$

As mentioned earlier, QP value is used by the video encoder to encode video data. In the context of SVBR, QP value produces  $R(k)$ . To obtain a certain  $R(k)$  value (bit

rate allocation), a suitable QP value should be determined. To calculate the next GoP QP value, which is  $Q(k+1)$ , SVBR requires  $R(k)$ ,  $q$ , and  $\varepsilon(x)$ . The relationship of all variables with the SVBR rate control block diagram is depicted in Figure 4.4.



**Figure 4.4:** Rate control block diagram with SVBR

The main idea here is as follows, if the current GoP produces high bit rate data (higher than  $r$ ) and the bucket fullness level is high as well (more than half of the bucket), a higher QP value for the next GoP should be generated. Higher QP value means lower video rate,  $R(k+1)$ , which is exactly what SVBR is supposed to produce. When the bucket is nearer to full level and VBR video sequence is active (high rate), the next video rate should be lower. Thus, the bucket does not tend to be full. The opposite will happen if the current video rate is low. The QP value will be reduced, which will produce higher video rate for the next transmission.

#### Case: SVBR Dynamic

How does Equation 4.8 and Equation 4.9 satisfy the above-mentioned principle? The explanation will be done using the following example;  $r = 20000$ ,  $R(k) = 16128$ ,  $Q(k) = 9$ ,  $q = 10$ ,  $X(k+1) = 17296$ , and  $b = 60000$ . In this case, at GoP- $k$ , SVBR video rate (which is  $R(k)$ ) is lower than CBR ( $r$ ). The idea is to increase the video rate for  $R(k+1)$  by reducing the QP value for  $Q(k+1)$ . By applying Equation 4.7,

$R_{open}(k+1)$  can be obtained as follows:

$$R_{open}(k+1) = \frac{R(k) \cdot Q(k)}{q} = \frac{16128 \times 9}{10} = 14515.2$$

Since  $R_{open}(k+1) < r$ , Equation 4.9 is employed. By applying Equation 4.5 to Equation 4.9, the Equation 4.9 can be written as follows:

$$Q(k+1) = \frac{q \cdot R_{open}(k+1)}{R(k+1)} \quad (4.10)$$

The values of  $\varepsilon(x)$  and  $R(k+1)$  can be obtained by using Equation 4.6 and Equation 4.5, respectively:

$$\varepsilon(x) = \frac{X(k+1)}{b} = \frac{17296}{60000} = 0.2883$$

$$R(k+1) = \varepsilon(x) \cdot R_{open}(k+1) + (1 - \varepsilon(x)) \cdot r$$

$$\begin{aligned} R(k+1) &= (0.2883 \times 14515.2) + ((1 - 0.2883) \times 20000) \\ &= 4184.7322 + 14234 = 18418.7322 \end{aligned}$$

Since  $R(k+1)$  higher than  $R_{open}(k+1)$ , Equation 4.10 will produce  $Q(k+1)$  which is lower than  $q$ , as shown below:

$$Q(k+1) = \frac{q \cdot R_{open}(k+1)}{R(k+1)} = \frac{10 \times 14515.2}{18418.7322} = 7.8807 \gg 8$$

It can be derived as well that in this case,  $Q(k+1)$  will be smaller than  $Q(k)$  by examining the the video rate level for  $R(k)$ ,  $R_{open}(k+1)$ , and  $r$ . Since  $R(k)$  which produces  $R_{open}(k+1)$  is smaller than  $r$ , and its equivalent QP ( $Q(k)$ ) is 9. To produce  $R(k+1)$  which is higher than  $R(k)$  then  $Q(k+1)$  should be smaller than  $Q(k)$ .



#### 4.2.2.3 The Implications of the Design

There are many implications from the original principles of the SVBR design. The implications are inherited from the way SVBR determines (predicts or estimates) a suitable bit rate allocation and quantization parameter value. From this estimation, the QP value will be fed into the video encoder (refer to Figure 4.4) to generate the real encoded video. The video rate for the encoded video may very much differ from the earlier video rate estimation, when estimating a suitable bit rate allocation.

The explanation to the above-mentioned implication is summarized in Table 4.2 and some related examples are shown in Table 4.3. The general idea of the SVBR algorithm is to estimate the QP value which then will be used by the video encoder to produce the encoded video sequence. The principle is to reduce the video bit rate when the previous video bit rate is high. To implement this principle, SVBR increases the QP value, which depends on the difference between  $R_{open}$  and  $r$ , and the bucket fullness level. The bigger the difference, the higher QP value is generated.

#### Case: Sensitive Relationship

The big difference between  $R_{open}$  and  $r$  creates a sensitive relationship. A small change might change the QP value to a much higher value than the previous one. For instance, in the case of  $r = 20000$ ,  $R(k) = 15120$ ,  $Q(k) = 25$ ,  $q = 10$ , and  $X(k+1) = 55120$ , the next QP value, which is  $Q(k+1)$ , is 18. The calculation of this result is as follows:

$$R_{open}(k+1) = \frac{R(k) \cdot Q(k)}{q} = \frac{15120 \times 25}{10} = 37800.0$$

Since  $R_{open}(k+1) > r$ , Equation 4.10 is employed. Estimation value for  $R_{est}(k+1)$ , as in Equation 4.4, is as below:

$$R_{est}(k+1) = (1 - \varepsilon(x)) \cdot R_{open}(k+1) + \varepsilon(x) \cdot r$$

$$\varepsilon(x) = \frac{X(k+1)}{b} = \frac{55120}{60000} = 0.9187$$

$$R_{est}(k+1) = ((1 - 0.9187) \times 37800) + (0.9187 \times 20000)$$

$$R_{est}(k+1) = 3074.4 + 18374 = 21448.4$$

$$Q(k+1) = \frac{q \cdot R_{open}(k+1)}{R(k+1)} = \frac{10 \times 37800}{21448.4} = 17.62 \gg 18$$

**Table 4.2:** Relationship of bucket fullness level, active sequence, and QP value

	$\varepsilon(x) < 0.5$	$\varepsilon(x) > 0.5$
$R_{open}(k) > r$	$Q(k+1) > q$  <i>The more <math>R_{open}(k)</math> and <math>r</math> differs, and the higher <math>\varepsilon(x)</math>, producing the higher <math>Q(k+1)</math>.</i>	$Q(k+1) > q$  <i>The more <math>R_{open}(k)</math> and <math>r</math> differs, and the higher <math>\varepsilon(x)</math>, producing the higher <math>Q(k+1)</math>.</i>
$R_{open}(k) \leq r$	$Q(k+1) \leq q$  <i>The more <math>R_{open}(k)</math> and <math>r</math> differs, and the higher <math>\varepsilon(x)</math>, producing the higher <math>Q(k+1)</math>.</i>	$Q(k+1) \leq q$  <i>The more <math>R_{open}(k)</math> and <math>r</math> differs, and the higher <math>\varepsilon(x)</math>, producing the higher <math>Q(k+1)</math>.</i>

Another big implication is on the encoded video rate. Since the estimated QP value is used to encode the video, and the next original video rate is much different from the previous one, this might create another undesired relationship. A numerical example will be shown in the next section.

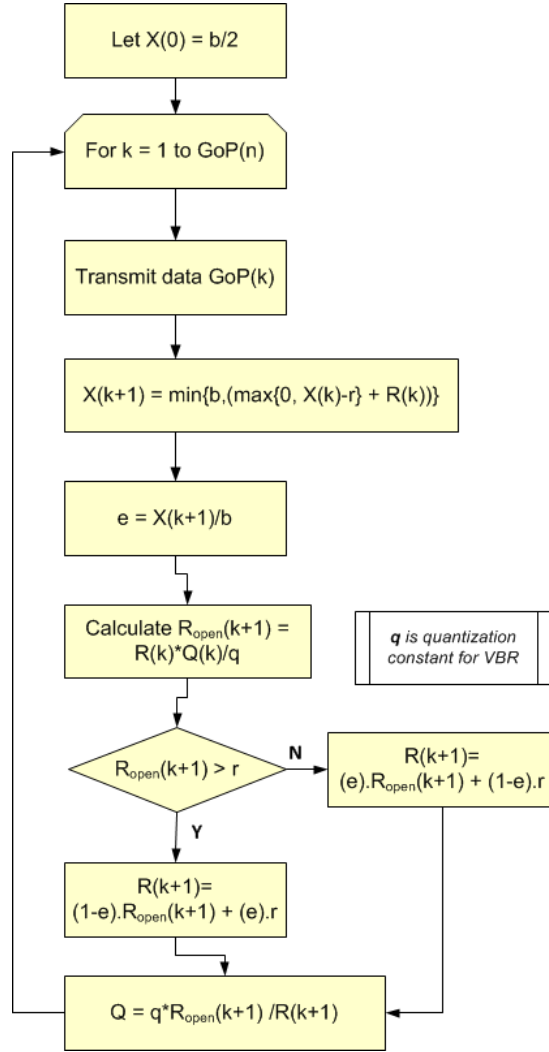
### 4.2.3 SVBR Algorithm Flow

The gross or important parts of the SVBR algorithm can be illustrated in a flow chart form as shown in Figure 4.5. The following is the explanation on the notation, commands, or variables used;

**Table 4.3:** Examples of the relationship of bucket fullness level, active sequence, and QP value

	$R_{open}(k+1)$	$r$	$\varepsilon(x)$	$q$	$Q(k+1)$
1	20160	20000	0.1667	10	10.0133 (10)
2	20160	20000	0.4167	10	10.0332 (10)
3	20160	10000	0.1667	10	10.9170 (11)
4	20160	10000	0.4167	10	12.6580 (13)
5	18144	20000	0.5833	10	7.8600 (8)
6	18144	20000	0.9167	10	9.4836 (9)
7	4500	20000	0.5833	10	2.9752 (3)
8	4500	20000	0.9167	10	6.7925 (7)

- $X(0)$ : is the initialization bucket fullness level at GoP-0, which is set to half of the bucket.
- For  $k$ : performs looping in order to process all video sequences from the first GoP to the last GoP.
- Transmit data: transmits each GoP data. The real implementation in the simulation package is much more complex. This is due to the fact that the simulation package imitates almost 100%, similar to the real network behavior. Thus, the implementation coding, that is presented in Chapter 6, needs to transform GoP data granularity into video frames then into transport datagrams.
- $X(k+1)$ : after the transmission, the virtual bucket fullness will be calculated.
- $e$ : represents  $\varepsilon(x)$ , which is the ratio of bucket fullness level to bucket size.
- Calculate  $R_{open}(k+1)$ : calculates the next GoP VBR data. As described previously, SVBR uses prediction and approximation in calculating the next GoP VBR data.
- $R_{open}(k+1) > r$ : tests whether VBR data is a higher video rate than  $r$ . From that test, SVBR can determine either the next GoP video rate is closer to  $R_{open}$  or  $r$ .



**Figure 4.5:** SVBR algorithm flow chart

- $Q$ : represents  $Q(k+1)$ , the quantization parameter for the next GoP video data.

### 4.3 Evaluation of Strengths and Weaknesses of SVBR

This section presents performance evaluation experiments of SVBR to analyze its strengths and weaknesses. The section also shows a performance comparison of SVBR to the traditional rate controllers, which are CBR and VBR.

### 4.3.1 The Evaluation Approach

A combination of several video clips, as discussed in the previous chapter, is used for the purpose of evaluating SVBR. These clips consist of low and high video rate, thus the SVBR performance in several scenarios can be observed.

### 4.3.2 Rate Control Experiment Settings

In order to execute the experiments, the following parameter setting is employed;

- Bucket size ( $b$ ) = 60000 Bytes. This size is equal to 1Mbps transmission speed.
- Leak rate ( $r$ ) = 20000 Bytes/GoP. This rate is average open-loop rate for the video sequences used, as suggested by Hamdi et al. in [23].
- VBR Initial  $q = 10$ . This initial value is selected to provide space for the SVBR rate control to increase or decrease QP value freely. For a more limited network resources setup, a higher initial  $q$  value should be considered.

The other settings have been discussed in length in Chapter 3.

### 4.3.3 The Result

The result of the experiments are shown in Figure 4.6, 4.7, 4.9, and in 4.10. Figure 4.6 and Figure 4.7 show the result of SVBR video rate. In Figure 4.6, the SVBR video rate is compared with VBR and CBR video rates, while in Figure 4.7, the SVBR video rate is charted with its respective bucket fullness level. Figure 4.9 presents PSNR values gained from the experiments for CBR, VBR, and SVBR video sequences. On the other hand, Figure 4.10 plots the QP values gained (from the estimation calculation) and used in producing CBR, VBR, and SVBR video rates.

In Figure 4.6, the SVBR video rate for GoP 201 is actually 253,000 Bytes/GoP. It has been reduced to 65000 for the purpose of chart readability and clarity. Several

other general observations can be noted as follows;

- A long flat scene can be observed from GoP 26 to GoP 200.
- A heavy VBR active sequence can be observed in GoP 369 to GoP 381. The video rates jump to around 65,000 Bytes/GoP.
- When the SVBR video rates are below  $r$ , the bucket fullness level is also relatively at a low level as well.
- Whenever the SVBR video rates are almost at 30000 Bytes/GoP, the bucket fullness level tends to be full.

In general, SVBR has demonstrated its novel creation. Its video rate is steadily somewhere between VBR and CBR. When heavy fluctuations occur, the SVBR video rate is interestingly, quickly adjusting itself into a permissible rate. However, it is obvious that SVBR fluctuates more than CBR and VBR. The analysis on this matter will be elaborated on the next section.

#### **4.3.3.1 Observation on the SVBR video rate and Bucket Fullness Level**

The long flat scene does not mean there is no movement in the video sequence. When the real GoP data sizes are examined deeply, as shown in Table 4.4, the sizes are not same. These small changes (in GoP data size) are due to minimum movement in the video scene or constant movement. The constant movement, even in the high video rate video frames, lead to minimum changes in the GoP data size. This is due to motion compensation video compression.

An additional reason for a long flat rate is on the approach that the transmitted video rates are calculated. For transmission purposes, TFRC in particular, the packet size is fixed equally for all packets, which is 1008 Bytes. It consists of 972 bytes of data payload and 36 bytes of header. Thus, for instance, a 50 bytes video frame will be

transmitted as 1008 bytes data per packet. As a consequence, two transmissions with different frame sizes might produce the same transmitted video rate, as illustrated in Figure 4.8.

In Figure 4.7, the bucket fullness level is charted with the SVBR video rate. For the first part, from GoP 1 to GoP 201 when there is no sudden jump in the SVBR video rate, the bucket fullness level is slightly higher than SVBR video rate. After the sudden jump occurs (at GoP 201), the bucket fullness level tends to be filled. The implication of this scenario will be elaborated in the next section.

After the second SVBR video rate sudden jump occurs, which is at GoP 369, the bucket fullness level starts to decrease. Specifically, it happens at GoP 381. At that time, the SVBR video rates also decrease from 57,456 Bytes/GoP to 34,272 Bytes/GoP and, then, for several consecutive GoPs it remains at the rate of around 18,000 Bytes/GoP. It can somewhat be concluded that the bucket fullness level will remain at a low level when the SVBR video rate is maintained at a low level. When the bucket starts to be at almost full level and the next scenes are active, the bucket fullness level tends to remain at full level as well. It will retract to a low level again when the next several GoPs video rate decreases to a low rate accordingly.

#### **4.3.3.2 Observation on the PSNR and QP Values**

In terms of PSNR value, as illustrated in Figure 4.9, SVBR gains a considerable good value until frame 3895. From that frame onwards, SVBR never obtains any good PSNR value until the end of the transmission. It might be as a result of several frame drops that occurred before and during that period. During several occasions, its PSNR value decreased badly, below 10dB. This can be seen clearly at the frame 2425, 4380-4401, and 5771-5821.

Figure 4.10 charts QP values for VBR, CBR, and SVBR rate controllers. As described in the previous chapter, the lower the QP value is, the higher the visual

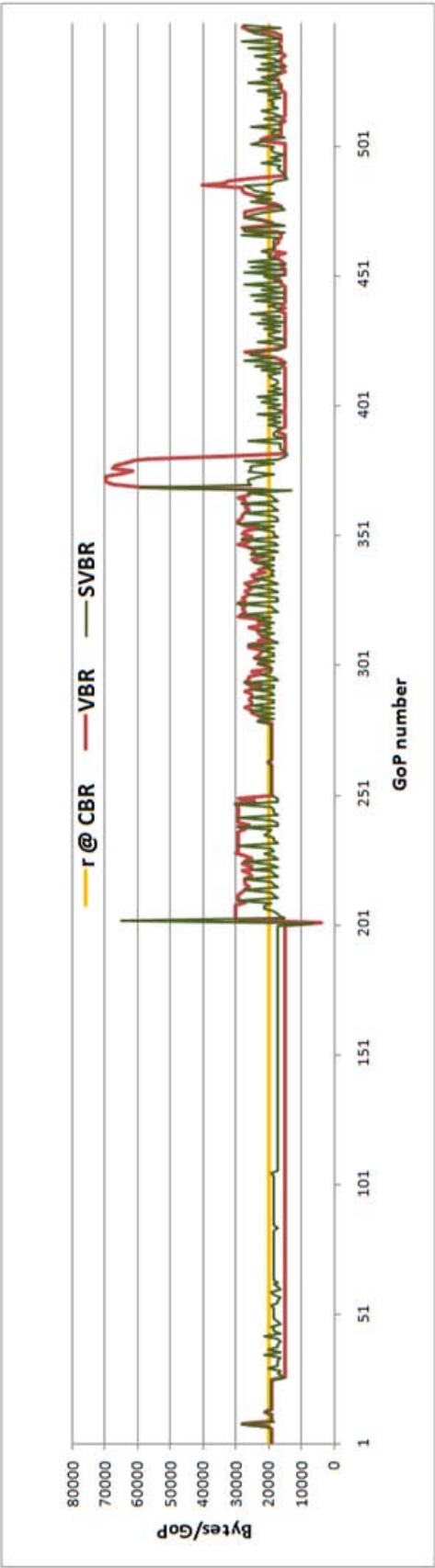


Figure 4.6: The SVBR experiments - comparing the result with VBR and CBR

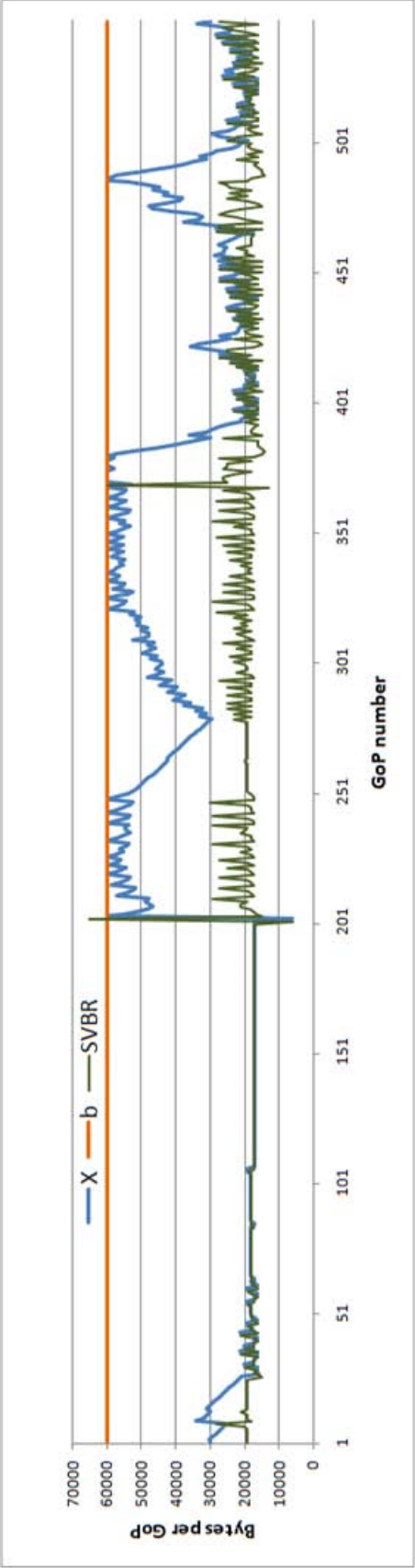
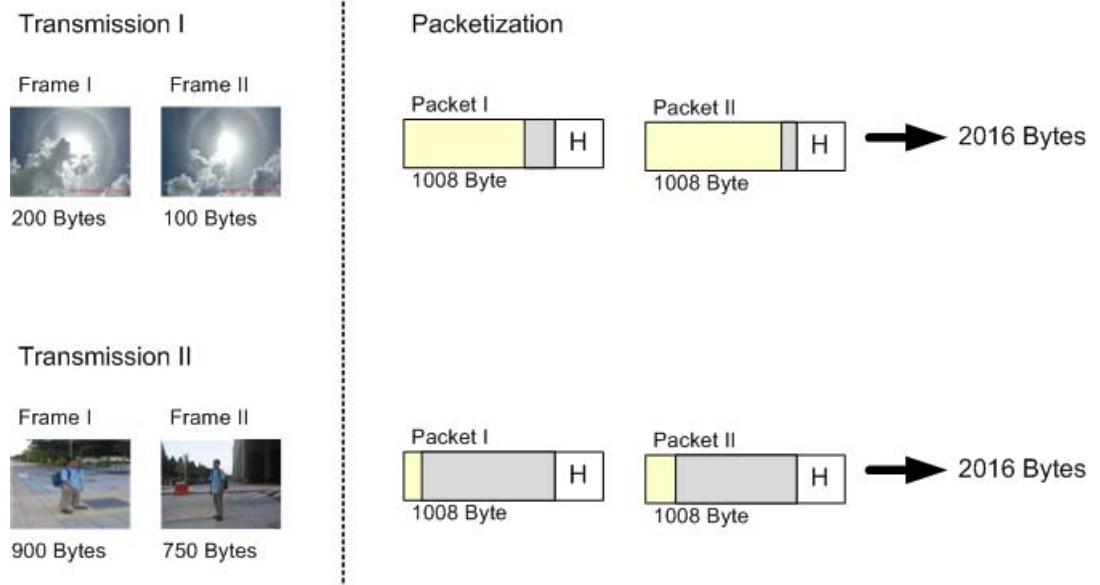


Figure 4.7: The SVBR experiments - bucket fullness level



**Table 4.4:** Data size for GoP 26-40

GoP #	video rate (Bytes/GoP)
26	4941
27	4932
28	4981
29	4986
30	5012
31	5003
32	4989
33	5007
34	4876
35	4939
36	4929
37	4969
38	4967
39	4967
40	4921



**Figure 4.8:** GoP data size calculation for TFRC transmission.

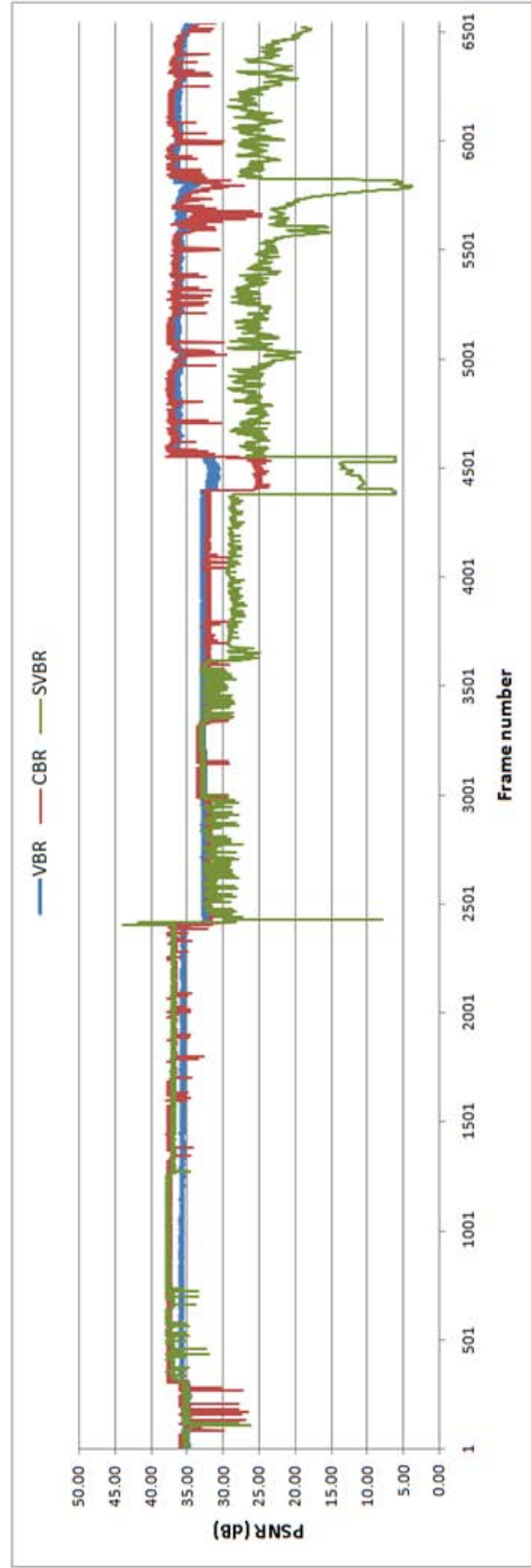
quality of the video and video rate will be. In addition, the more constant the QP value is, the more constant the video visual quality. From the chart, on the average SVBR gains lower QP value, but it is less constant.

In order to generate an equivalent PSNR and QP values for the CBR rate controller, the video sequence needs to be executed in the similar system. Since the system is developed based on GoP granularity, the exact constant video rate is impossible to be achieved. However, since the main objective of this study is to create a new rate control algorithm based on SVBR algorithm, the main performance comparison is between SVBR and the new algorithm.

Hence, a CBR rate controller is created, even though it is not really constant, it still uses the nearest GoP value with a constant video rate. Thus, it still can be used as another comparison. Accordingly, it should be highlighted that in the case of CBR and SVBR in acquiring constant QP value, SVBR provides much better constant video visual quality than CBR rate controller. The real generated the CBR video rate is shown in Figure 4.11.

#### **4.3.4 The Analysis**

After describing some observations that can be made from the experimental result, the extensive analysis will be elaborated on in this section. The strengths and weaknesses of the SVBR algorithm will be highlighted based on the analysis of several scenarios below. The analysis will be based on how SVBR reacts when certain scenarios occur. The analyzed scenarios include the occurrence of sharp decrease in VBR video rate, sudden bucket overflow, low video rate with low bucket fullness level, fluctuating video rate, and SVBR with varied video rates.



**Figure 4.9:** The SVBR experiments - PSNR value

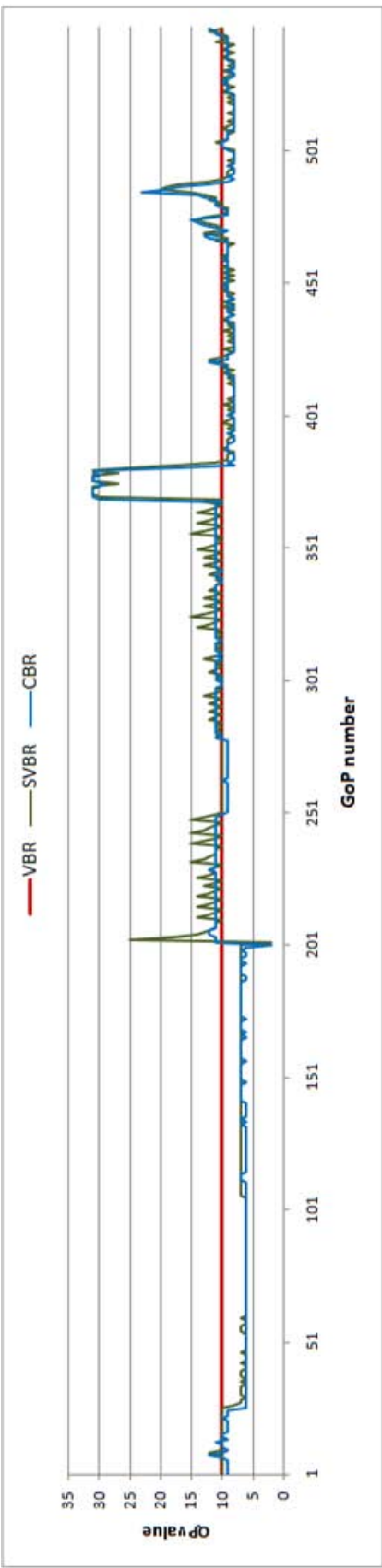


Figure 4.10: The SVBR experiments - QP values

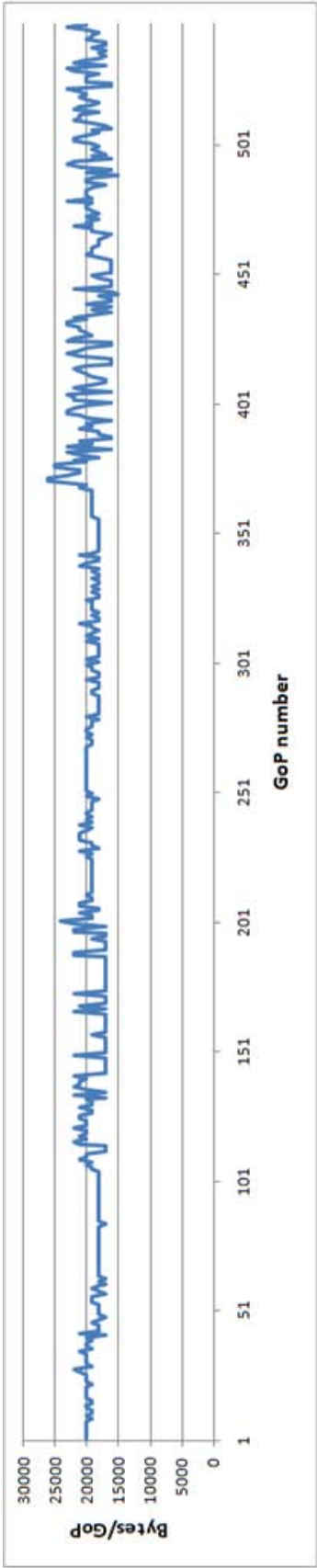


Figure 4.11: Emulating CBR with GoP video rate granularity

### Case 1: Sharp Decreases in SVBR video rate

Figure 4.12 shows the phenomenon of sharp decrease in SVBR video rate which can be seen at the GoP-201. Figure 4.12 charts the data for this particular GoP and the surrounding GoPs. The sudden sharp decrease in SVBR video rate is from 17,136 Bytes/GoP to 6,048 Bytes/GoP. The SVBR video rate is examined here because the generation of the next SVBR rate is very much dependent on it (refer to Figure 4.5);  $R_{est}(k+1)$  is the SVBR video rate estimation for the GoP- $(k+1)$ .

What can be observed here is that when a sharp decrease in SVBR video rate occurs, it will automatically generate a sudden burst in the next SVBR video rate. This kind of scenario occurs especially when the next VBR video rate is increased sharply as well. A possible problem of this scenario is that the SVBR video rate might be increased at a very much higher rate as compared to VBR. This case is clearly seen in Figure 4.12a. The SVBR video rate has bursted from 6048 Bytes/GoP to 253000 Bytes/GoP, while, the VBR video rate has only increased sharply from 4032 Bytes/GoP to 30240 Bytes/GoP.

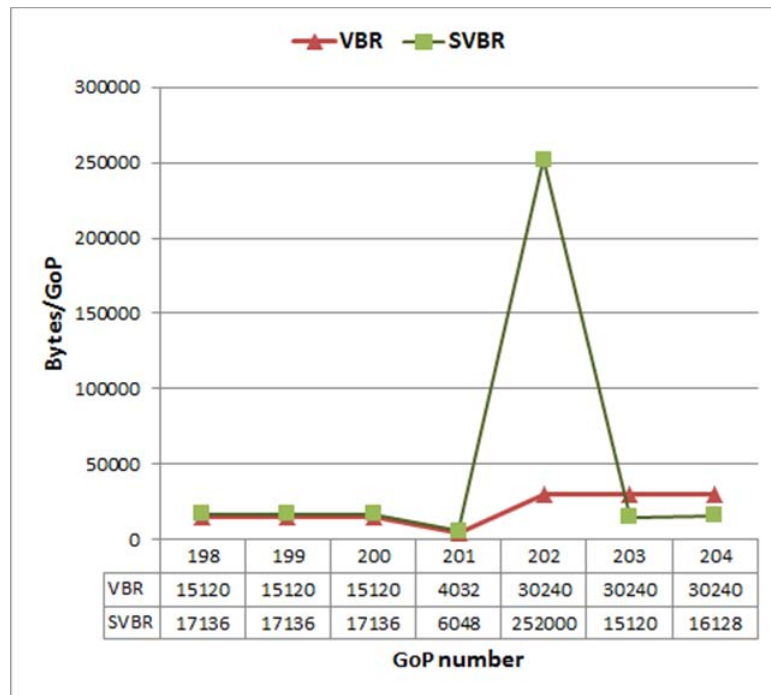
This above scenario can be explained as follows:

For Gop-201, the  $k$  is equal to 201. Then, the bucket fullness ratio for the next GoP, when applying Equation 4.6, is 0.1008. The QP values for the respective GoPs are shown in the Table 4.5.

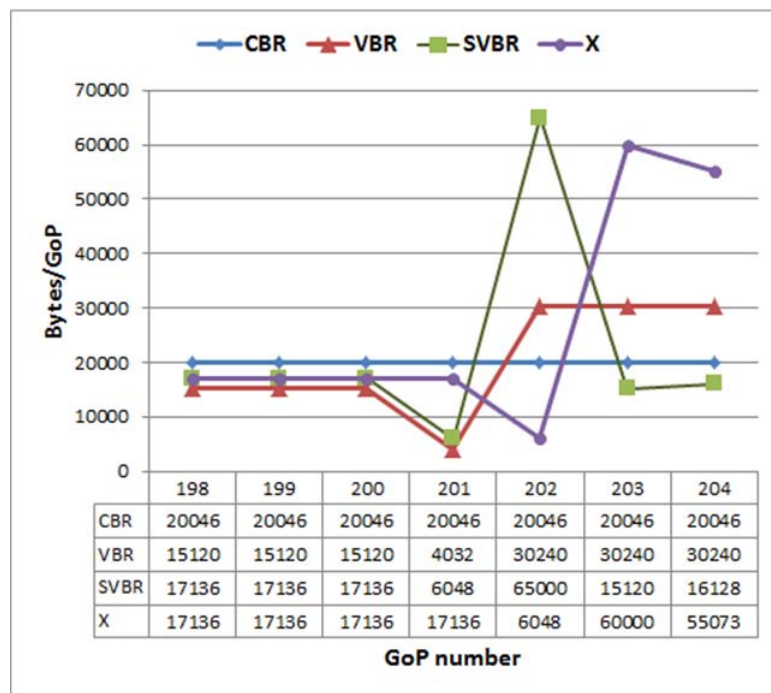
**Table 4.5:** Respective QP values for GoP 198-204

GoP #	198	199	200	201	202	203	204
QP	7	7	7	7	2	25	18

$$\varepsilon(x) = \frac{X(202)}{b} = \frac{6048}{60000} = 0.1008$$



(a) Real video rate



(b) Modified video rate

**Figure 4.12:** Sharp decrease scenario in the GoP-201

$$R_{open}(202) = \frac{R(201) \cdot Q(201)}{q} = \frac{6048 \times 7}{10} = 4233.6$$

Since  $R_{open}(202)$  is less than  $r$  (CBR=20000), then Equation 4.5 is applied;

$$Q(202) = \frac{q \times R_{open}(202)}{(\epsilon(x) \times R_{open}(202)) + ((1 - \epsilon(x)) \times r)}$$

$$Q(202) = \frac{10 \times 4233.6}{(0.1008 \times 4233.6) + (0.8992 \times 20000)}$$

$$Q(202) = \frac{42336}{426.7469 + 17984} = \frac{42336}{18410.7469} = 2.2995$$

Since the nearest QP value for  $Q(202)$  is 2, then  $Q(202) = 2$ . By referring to Table 4.6, the video rate for QP=2 for GoP-202 is 252,000 Bytes/GoP. What should be highlighted here is that, when at GoP-201, the video rate is comparatively very low, so the algorithm wants to increase the video rate for the next GoP by lowering the QP value. The algorithm succeeds in its principle to perform that task. The QP value was decreased from 7 to 2, with the intention to increase the video rate substantially. However, at GoP 202, VBR video rate increases to a higher rate, consequently QP=2 for SVBR GoP 202 video rate is equal to 252000 Bytes/GoP. For reference purpose, if VBR video rate remains at the same rate for GoP-202 as the GoP-201, QP=2 will moderately increase SVBR GoP 202 video rate to 24192 Bytes/GoP.

**Table 4.6:** Respective QP and video rate values for GoP-202

$Q(202)$	2	3	4	5	6	7	8
<b>video rate (KBytes/GoP)</b>	252.0	149.2	998.0	735.8	483.8	463.7	332.6

From the above calculation, it can be concluded that one of the weaknesses of SVBR algorithm is when a sharp decrease in VBR video rate occurs, the SVBR video rate for the next GoP will increase dramatically. This is especially true in the case that

the next VBR video rate has increased sharply as well. The sudden sharp fluctuation in the SVBR video rate can go up beyond the permissible level, which overflows the bucket. This scenario crops up as a result of very low bucket level fullness and low  $R_{open}$  value case. This leads to a very low QP value and consequently, produces a very high video rate.

### Case II: When a Sudden Bucket Overflow Occurs

Figure 4.13 shows the case when a sudden bucket overflow occurs. This scenario can be observed starting from GoP 369 to 381. This phenomenon can be described as an advantage of the SVBR algorithm, but at the same time, it shows poor performance as well. The disadvantage of this algorithm can be seen at GoP-369. Here, when VBR video rate fluctuates sharply to 59472 Bytes/GoP, exceeding the bucket size, the SVBR also follows this behavior, even to higher video rate than the VBR.

This scenario can be shown as follows:

$$\varepsilon(x) = \frac{X(369)}{b} = \frac{53104}{60000} = 0.8851$$

$$R_{open}(369) = \frac{R(368) \cdot Q(368)}{q} = \frac{13104 \times 13}{10} = 17035.2$$

Since  $R_{open}(369)$  is lower than  $r$ , then Equation 4.9 is applied:

$$Q(369) = \frac{q \times R_{open}(369)}{(\varepsilon(x) \times R_{open}(369)) + ((1 - \varepsilon(x)) \times r)}$$

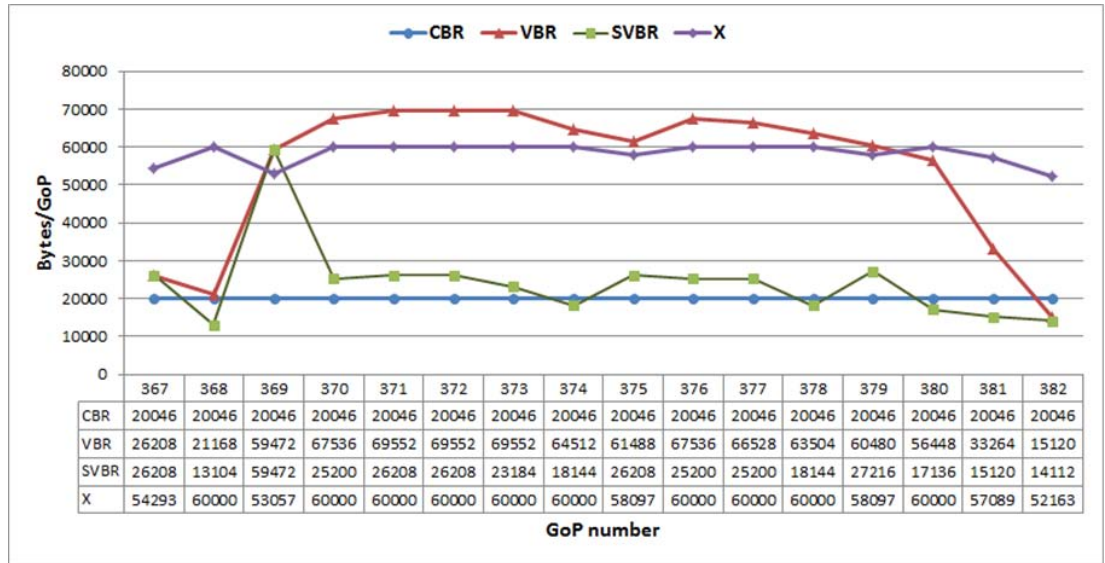
$$Q(369) = \frac{10 \times 17035.2}{(0.8851 \times 17035.2) + (0.1149 \times 20000)}$$

$$Q(369) = \frac{170352}{15077.8555 + 2298} = \frac{170352}{17375.8555} = 9.8039$$



Since the nearest QP value for  $Q(369)$  is 10, then  $QP = 10$ . Then, accordingly the whole transmission video rate for  $QP=10$  is 59472 Bytes/GoP. This phenomenon is almost similar to the phenomenon in the previous subsection.

In addition to the aforementioned explanation, since SVBR uses previous GoP data, which is GoP 368, to estimate the next GoP data, it generates a small  $R_{open}$ . The small  $R_{open}$  and higher bucket fullness level will generate  $R(369)$  (refer Equation 4.5) close to the  $R_{open}$  value (refer to Table 4.1). Consequently, the SVBR algorithm generates a QP value which is close to the default value  $q$ . This form of estimation is good for video sequences which are increased or decreased in a smoother way, but, for the next video sequence, even though the video rate has sharply increased, the  $q$  value is still the same as for previous video sequence. Then, for SVBR, when the QP value is around the default  $q$  value, it produces a high video rate. This is due to the fact that SVBR uses real data for the transmission, which is same as VBR ( $Q(369)=q=10$ ).



**Figure 4.13:** Sharp VBR increase scenario

The advantage of SVBR algorithm for this scenario can be observed by how SVBR reacts after any sudden sharp increases, specifically, after GoP 369 until GoP 380. For that period, even though the VBR video rate is maintained above the bucket level,

SVBR quickly adjusts itself to a permissible level.

This scenario can be shown as follows:

$$\varepsilon(x) = \frac{X(370)}{b} = \frac{60000}{60000} = 1.0$$

$$R_{open}(370) = \frac{R(369) \cdot Q(369)}{q} = \frac{59472 \times 10}{10} = 59472$$

Since  $R_{open}(370)$  is greater than  $r$ , then Equation 4.8 is applied:

$$Q(370) = \frac{q \times R_{open}(370)}{((1 - \varepsilon(x)) \times R_{open}(370)) + (\varepsilon(x) \times r)}$$

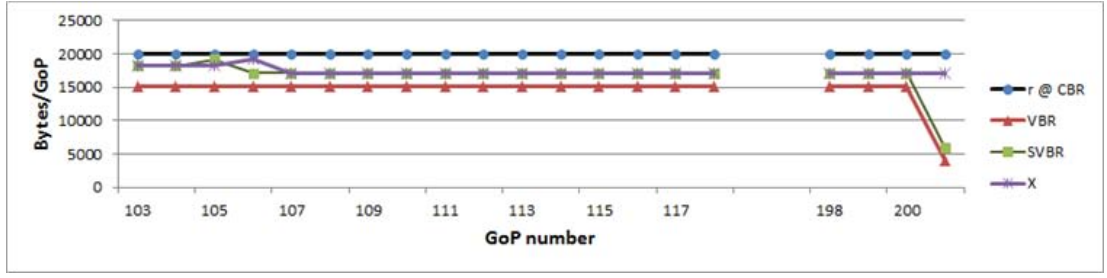
$$Q(370) = \frac{10 \times 59472}{(0 \times 59472) + (1 \times 20000)}$$

$$Q(370) = \frac{594720}{20000} = 29.736$$

Since the nearest QP value for  $Q(370)$  is 30, then  $QP = 30$ . The video rate for QP=30 is 25200 Bytes/GoP, then accordingly the SVBR video rate for GoP-369 is 25200 Bytes/GoP.

In the above case, although the present VBR and estimation of the next VBR ( $R_{open}(370)$ ) are high, and since the bucket fullness is also high, the estimation for the  $R(370)$  is getting closer to  $r$  or CBR. All this will produce a higher QP value ( $Q(370)$ ) than  $q$ . Accordingly, the higher QP value will generate a low SVBR video rate. This is a good strength of the SVBR algorithm. Even though the VBR video rate is very high and it is maintained high for several consecutive GoPs, the SVBR algorithm quickly adjusts its video rate to an acceptable level and maintains that rate onwards.

It can be concluded here that when a high video rate burst occurs, for the next first GoP, SVBR will burst as well leading to bucket overflow. As each GoP consists of



**Figure 4.14:** A long-low-flat video rate sequence

12 frames, this means that 12 frames will burst as well. However, SVBR has been designed to quickly retract the burst to a permissible level for the following GoPs.

### Case III: Low video rate and Low Bucket Fullness Level

As shown in Figure 4.14, this scenario can be clearly seen from GoP 26 to 200. From this range, especially from GoP 106 to 200, it shows a long flat low rate sequence. Part of this sequence is shown in Figure 4.14. Here, both of CBR and VBR are at low rate, SVBR is seen to be around the both rates. SVBR should increase its video rate here to improve its visual quality. It clearly shows here that SVBR is not designed to increase the visual quality in a low video rate sequence.

Based on Figure 4.14, for SVBR to estimate GoP 106 from GoP 105, SVBR bases on its distance to CBR ( $r$ ) and VBR. In the case of calculating GoP 106 using GoP 105, where the video rates are very close to  $r$ , this forces SVBR to adjust its rate to lower than  $r$  value. It does that by lowering the  $R(106)$  estimation from the real value of the  $R(105)$  (refer to equation 4.5). By designing in such a way, it produces a higher QP value estimation, which is 7 (previously QP=6). Consequently, it generates lower value of video rate for the SVBR GoP 106 video rate. The calculation of this algorithm can be inspected as follows:

$$\varepsilon(x) = \frac{X(106)}{b} = \frac{19152}{60000} = 0.3192$$

$$R_{open}(106) = \frac{R(105) \cdot Q(105)}{q} = \frac{19152 \times 6}{10} = 11491.2$$

Since  $R_{open}(106)$  is lower than  $r$ , then Equation 4.9 is applied:

$$Q(106) = \frac{q \times R_{open}(106)}{(\epsilon(x) \times R_{open}(106)) + ((1 - \epsilon(x)) \times r)}$$

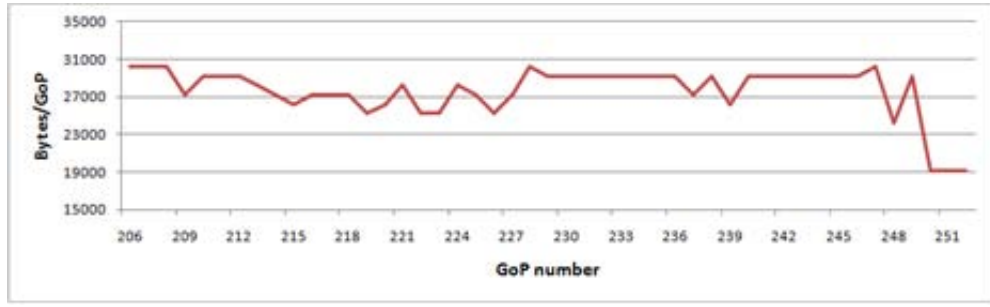
$$Q(106) = \frac{10 \times 11491.2}{(0.3192 \times 11491.2) + (0.6808 \times 20000)}$$

$$Q(106) = \frac{114912}{3367.9910 + 13616} = \frac{114912}{16983.9910} = 6.7659$$

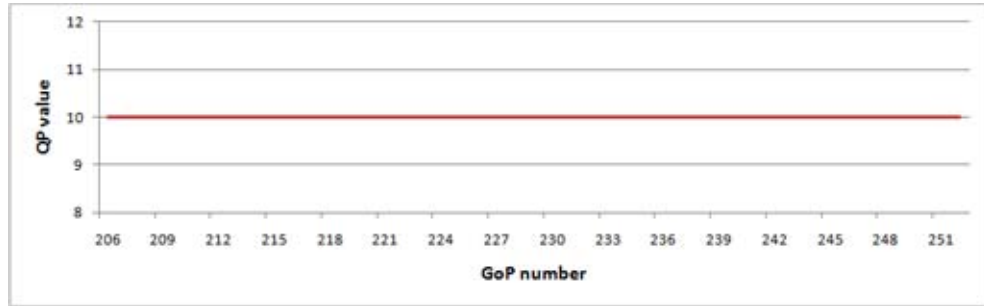
Since the nearest QP value for  $Q(106)$  is 7, then  $QP = 7$ . Then, accordingly the whole transmission video rate for QP=7 is 17136 Bytes/GoP.

In producing SVBR video rate for GoP 107, the SVBR will estimate  $R(107)$  as a little bit higher than the real  $R(106)$  since  $R(106)$  is located in the middle of the  $r$  and VBR but at a little bit lower distance. As a consequence, when calculating the value for QP, it gains  $Q(107)=6.7717$ . Since, the nearest round number for 6.7717 is 7, it produces the SVBR video rate for GoP 107 similar to the previous GoP. This scenario continues until GoP 200, where the VBR video rate for GoP changes to another value.

Therefore, the clear disadvantage of SVBR algorithm is that it has been designed to maintain its video rate to be vary between VBR and CBR video rate. Inevitably in a low video rate, especially in a long-flat video rate, SVBR tends to be around that rate. Thus, it maintains its visual quality. However, since the bucket fullness is at a low level, the video rate can be increased so that the visual quality can be increased as well.



(a) Positive video rate



(b) Constant QP value

**Figure 4.15: Positive fluctuation**

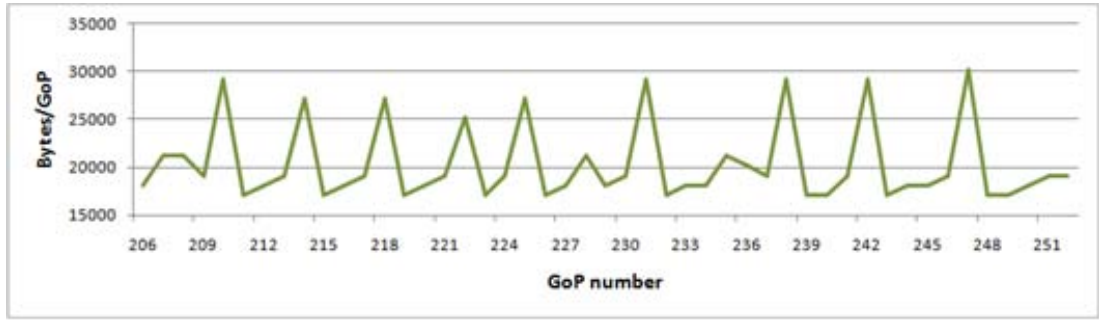
#### **Case IV: Fluctuating video rate**

There are two types of conflicting fluctuated video rates; positive and negative fluctuation. video rate fluctuates generally. A positive fluctuation is a condition where its QP values are constant. Thus, video rate fluctuation is considered as a positive fluctuation. Figure 4.15 illustrates this positive fluctuation condition.

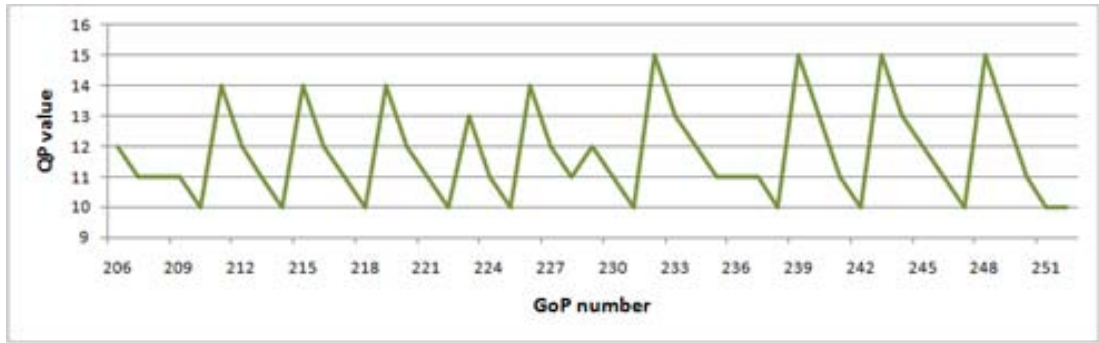
A negative fluctuation is a condition where the video rate and QP values are not constant. This negative fluctuation video rate condition is shown in Figure 4.16. This subsection refers to the negative fluctuation, especially the fluctuation in the SVBR video rate. In the following, two cases of the negative fluctuation are explained.

#### **Case IV-A: VBR is Higher than CBR**

From Figure 4.6, it can be observed that SVBR creates significant non-positive fluctuations. Fluctuation is positive as in VBR. Thus, wherever SVBR fluctuates differently from VBR, it can be considered as a negative fluctuation. One of this



(a) Negative video rate



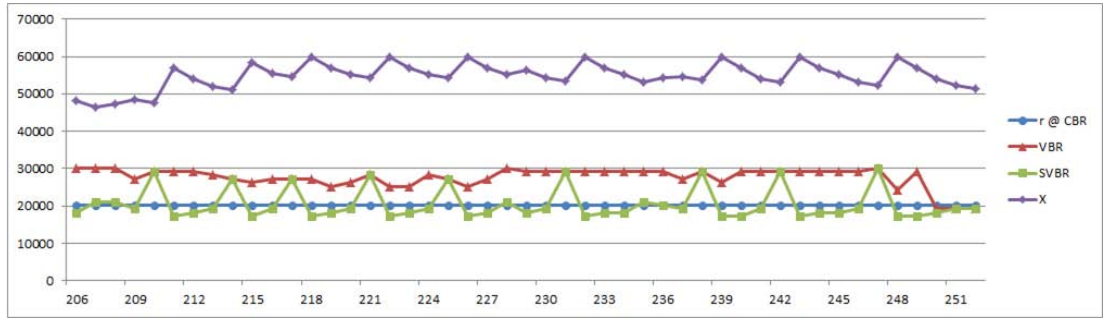
(b) Varying QP value

**Figure 4.16:** Negative fluctuation

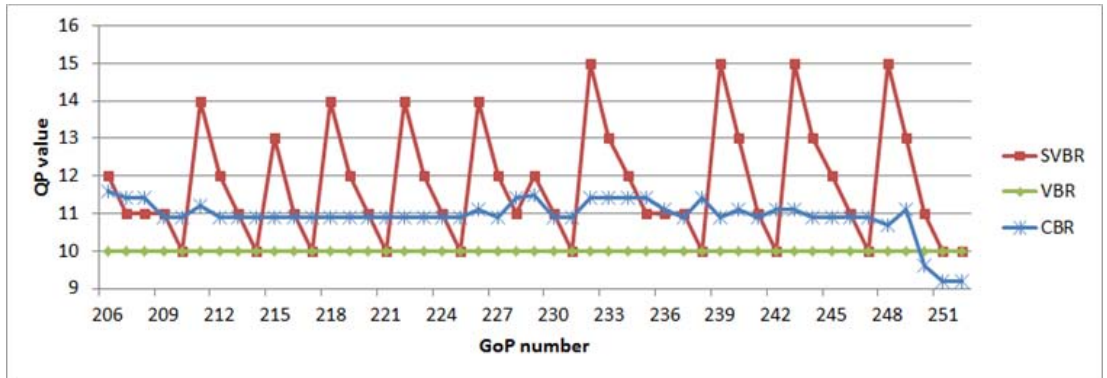
scenarios can be seen at GoP 208 to 250. Figure 4.17 shows this scenario.

This scenario is one of the good examples of the sensitive relationship as describe in Subsection 4.2.2.3. In the GoP 209,  $QP=11$  and SVBR video rate ( $R(209)$ ) is 19152. This video rate is below both CBR and VBR video rates at that GoP. One of the principles of SVBR is to let the next video rate lie between VBR and CBR video rates by adjusting the next QP value. It should be emphasized here that SVBR is only able to estimate the next QP value with the hope that the actual video rate will be lying between the intended region.

Since the generated  $R_{open}$  for GoP 210 is a little bit higher, it will produce a little higher QP value (refer to Table 4.2 and Table 4.3) but below 11. Another way to comprehend this scenario is by looking at how SVBR tries to increase its video rate to be between CBR and VBR. Given that the current video rate is less than both CBR and VBR rates with its QP value at 11, SVBR will decrease its QP value to increase its



(a) SVBR video rate fluctuation



(b) SVBR QP fluctuation

**Figure 4.17: SVBR fluctuation**

video rate, but to the maximum of 10. This is due to the fact that the QP value for the VBR is 10. The calculation of  $Q(210)$  gives 10.4223 and since there is no fraction of QP encoding, SVBR obtains  $Q(210)=10$ .

The  $Q(210)=10$  will generate a similar video rate with VBR GoP-210 video rate. Since, VBR video rate is in active sequence, SVBR obtains much higher video rate compared to the previous video rate. The negative implication of this situation is on the generation of the next QP and its video rate. As a result of the wider gap between  $R_{open}$  and  $r$ , it causes the generation of QP to become sensitive. Again, as described in Table 4.2 and Table 4.3, when the gap is wider and the bucket fullness level is almost full, it generates a much higher QP value. In the case of  $Q(211)$ , SVBR obtains 14.3137 (or 14 only). This QP, when translated to the actual video rate, is equal to 17136 Bytes/GoP. It creates a significant oscillation in the SVBR video rate. This situation continues until GoP 250 when the VBR video rate faces a less active scene.

#### **Case IV-B: CBR is Higher than VBR**

Similar conditions can be seen in other areas. At GoP 424 to GoP 448, similar negative fluctuation can be observed but this time the VBR is lower than CBR. Another dissimilar condition is that the bucket fullness level is also fluctuating between above and below the CBR video rate. Figure 4.18 illustrates this scenario.

The fluctuation occurs also as a result of sensitivity in the algorithm when the gap between VBR and CBR is quite wide. Although it is less wider as compared to the previous subsection, the changes in one QP value has contributed to this fluctuation. Another contribution factor to this fluctuation is the oscillation of the bucket fullness level at the CBR video rate.

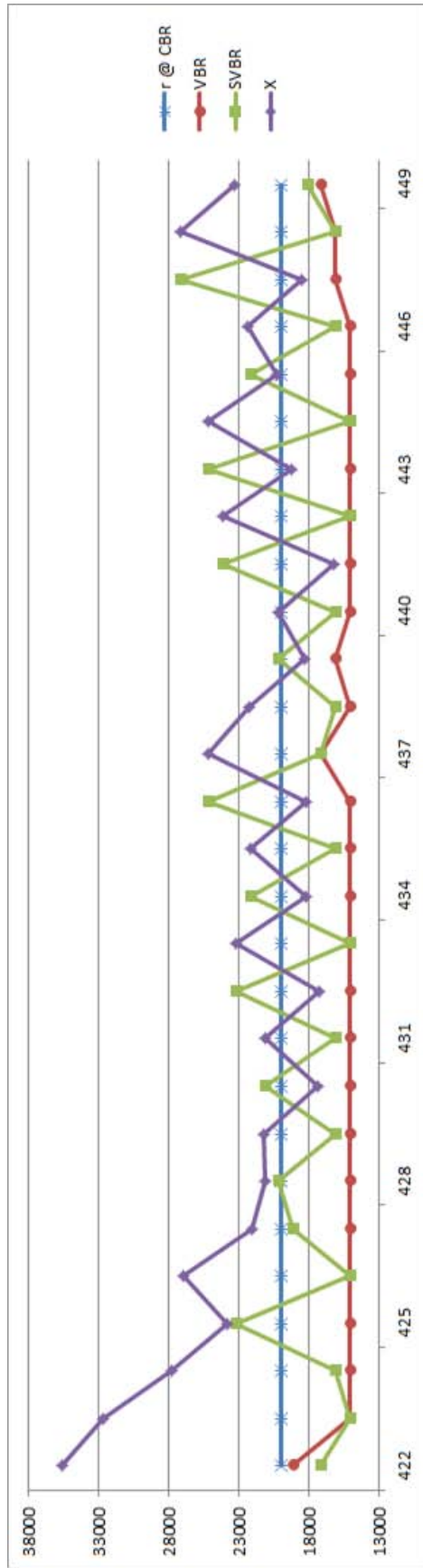
#### **Case V: SVBR with Varied video rates**

In theory, when SVBR is behaving almost similar to VBR video rate, the QP value will be almost as consistent as VBR video rate. Usually, in substantially varied video rates, the SVBR performs better regarding following the VBR changing video rate. This scenario can be observed starting at GoP 467 to 490, as shown in Figure 4.19.

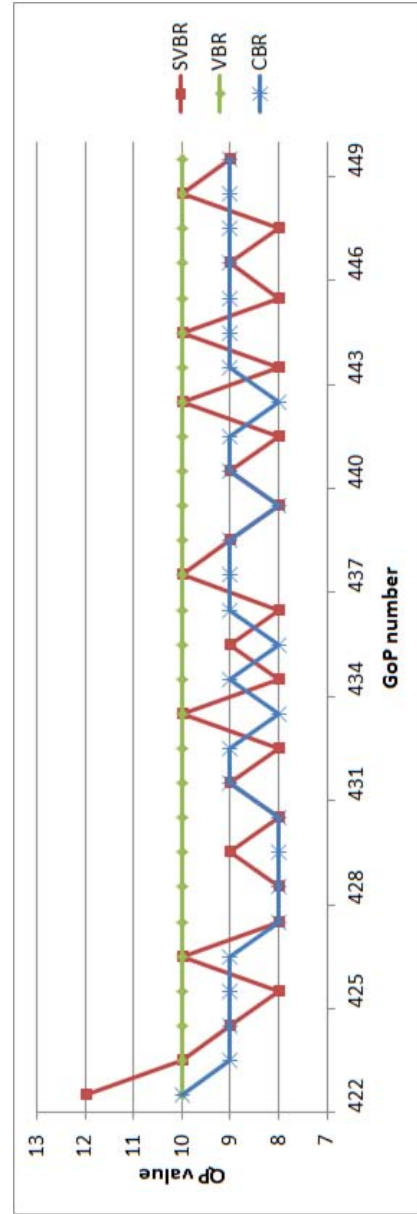
However, it does not reflect well in certain conditions. Figure 4.20 illustrates this circumstance when the associated QP values are charted. When in theory the SVBR QP values should follow VBR QP values, but instead, it seems SVBR is following CBR QP values.

This situation arises because of a similar reason as stated in the previous subsection. In a condition of high bucket utilization or wide gap between VBR and CBR, it has created a sensitive relationship between two consecutive QP values. Consequently, a small change in QP value leads to a fluctuated video rate.



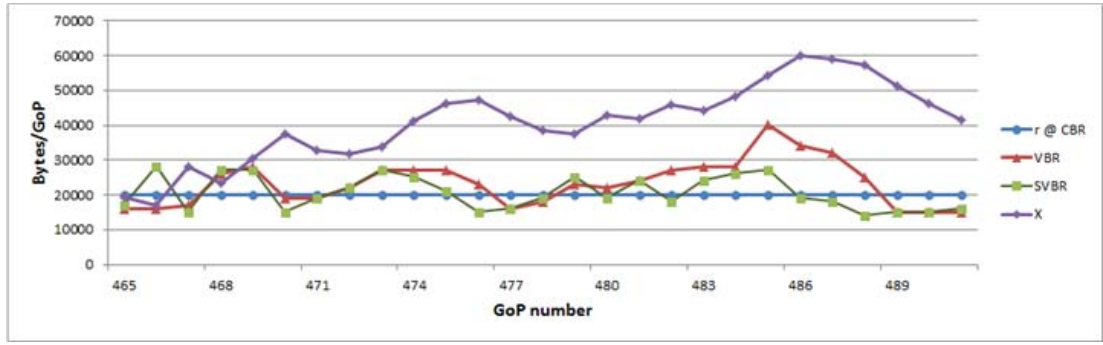


(a) SVBR video rate fluctuation



(b) SVBR QP value fluctuation

**Figure 4.18:** Negative SVBR fluctuation with VBR lower than CBR

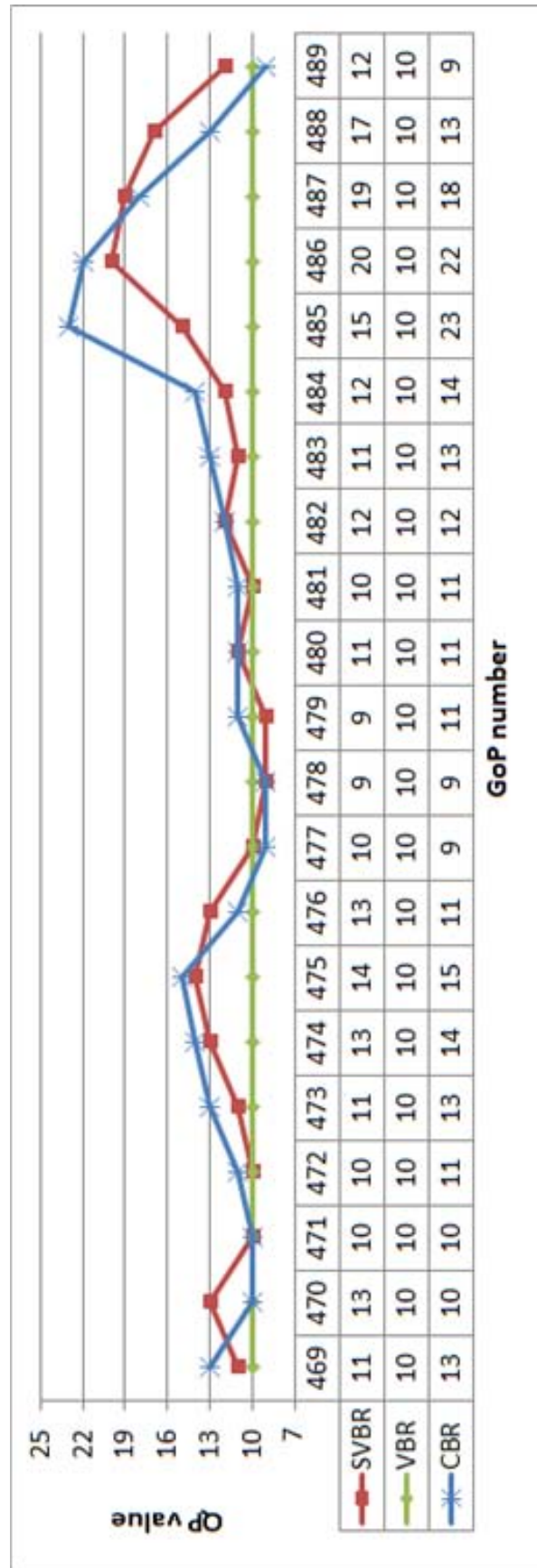


**Figure 4.19:** SVBR with varied VBR video rate

## 4.4 Summary

This chapter presented the extensive analysis of the SVBR algorithm performance. In general, this algorithm has demonstrated its novel creation of an ideal rate controller especially for real-time video application which produces a higher visual quality video, the video rate is within a permissible bandwidth level, and less delay as a result of no introduction of an additional buffer or complex computational algorithm.

However, there are spaces for performance improvement of the SVBR algorithm, especially under certain specific conditions. Besides the strengths of SVBR algorithm, this analysis had pointed out several weakness areas that can be considered for improvement. The circumstances that had been strongly stressed were when there is an occurrence of a sudden sharp decrease in the VBR video rate, the occurrence of a sudden bucket overflow, the existence of a low video rate with low bucket fullness level, and the generation of a cyclical negative fluctuation. These weaknesses play an important role in designing a new algorithm for video transmission application. In the next chapter, the design of a new algorithm based on this SVBR-based algorithm is presented.



**Figure 4.20:** Associate QP values for the varied VBR video rate

# **CHAPTER FIVE**

## **DESIGN AND IMPLEMENTATION OF A NEW SHAPED CONTROL ALGORITHM FOR A VIDEO APPLICATION**

In Chapter 4, the empirical evaluation of the SVBR algorithm has been extensively analyzed. The comprehensive discussion has identified the strengths, limitations, and weaknesses of the algorithm and has highlighted the important parts which can be enhanced. From that discussion, this chapter will present the design of the new shaped algorithm for a video application, which will be called SD-SVBR. The elaboration is on the design justifications, specifications, requirements, and objectives for the new algorithm. Then, the detailed design is discussed thoroughly.

### **5.1 Design Objectives**

Therefore, in creating a new video rate shaped algorithm, which is called SD-SVBR - a slight delay Shaped VBR, several design objectives have been considered. This new algorithm has the following features: (i) no sudden unwanted sharp increment or decrement in the video rate, (ii) high video rate, and (iii) a smoother negative fluctuation.

Besides the above mentioned features, this new algorithm will maintain the

SVBR's positive elements, i.e. the leaky bucket algorithm, which has helped in avoiding overflow by controlling the video rate. This algorithm is developed at GoP video granularity level, similar to the SVBR algorithm and many other rate control algorithms, such as in [192, 163, 166].

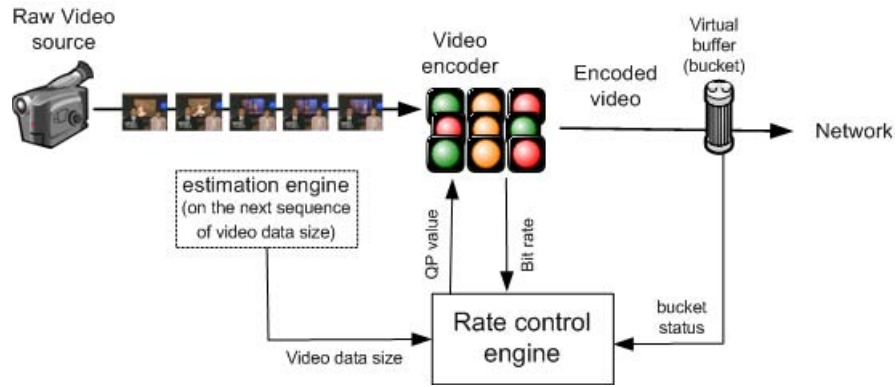
Figure 5.1 demonstrates the difference between SVBR and SD-SVBR algorithm. SD-SVBR attains its great advantages by buffering a GoP of the video sequence. In Figure 5.1a, no future information of the video sequence is used, thus it can be applied in the real-time video application. On the other hand, as depicted in Figure 5.1b, some advance video information are obtained thus it has to limit the application into a slight delay video transmission system. To be more specific, the slight delay here is a buffer of one GoP of video sequence, which is around 0.4 seconds (one GoP = 12 frames and video rate is 12 frames per second). Therefore, to apply the system in a stored video application, it can be employed easily. This is because in the stored video application, many video information can be obtained in advance.

## **5.2 SD-SVBR Algorithm**

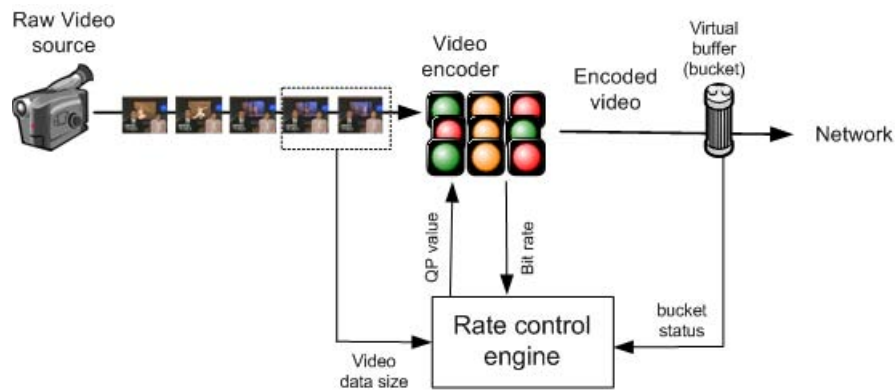
As described previously, the new algorithm has features such as no unwanted sharp video rate increment or decrement, a high video rate algorithm, and a smoother QP stability. These have contributed to a new shaped video rate which is superior to the SVBR. In the following subsections, an extensive discussion is presented in order to describe how the design objectives of SD-SVBR algorithm are achieved.

### **5.2.1 Avoiding Unwanted video rate Increment/Decrement by Using Real QP-to-video rate Matching**

Previously, it has been described that the fundamental contributor for the weaknesses and limitations of the SVBR algorithm are the employment of estimation and prediction methods in generating the SVBR video rate. Thus, the fundamental



(a) SVBR algorithm



(b) SD-SVBR algorithm

**Figure 5.1:** Working differences between SVBR and SD-SVBR algorithm

improvement is to avoid, as much as possible, the estimation and prediction in determining the bit rate allocation and its respective QP value.

However, it is not always possible to accomplish that principle in a real-time video application. Firstly, some forms of estimation or prediction have to be implemented because it is not always possible to obtain advanced information. This is true in the case of real-time or slight delay video application. Secondly, in particular for the leaky-bucket algorithm implementation, any allocation of the bit rates has implications to a certain extent.

For instance, by allocating a high bit rate for the next video sequence, it leaves a very small space in the bucket. As a consequence, only a small video rate can be allocated for the following video sequence. Thus, not only it produces a low quality

video sequence, but contributes to the fluctuation in the video rates for the consecutive video sequences.

Estimating for the middle video rate is not always good as well. Besides, it does not exploit the available bandwidth at the maximum capacity, and it also tends to be at a low video rate if both video rate references (CBR and VBR) are at the low rate. Thus, the new algorithm has been designed to utilize the available bandwidth but at the same time it reduces the unwarranted fluctuations.

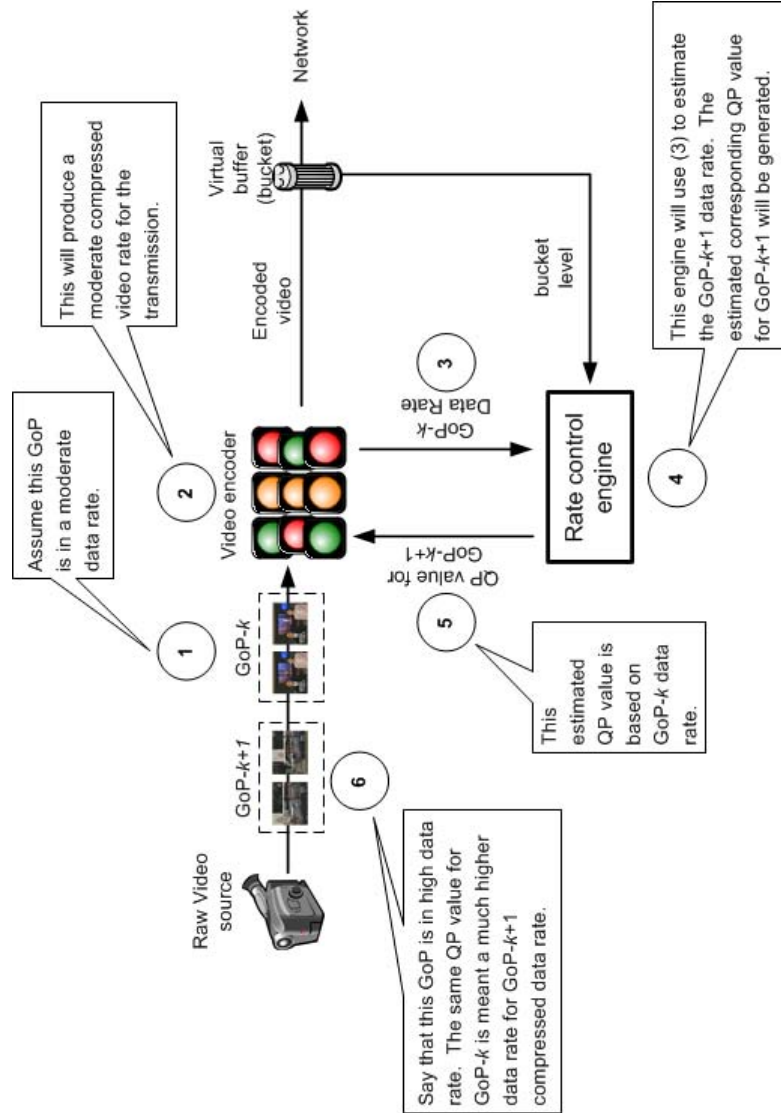
In short, the determination the QP value for a certain intended video rates can be obtained by analyzing the video sequence in advance. In a slight delay video application, the next GoP information can be programmed to make it available beforehand.

#### **5.2.1.1 Justification for Producing Correct QP and video rate Value**

By referring to Figure 4.4, the correct corresponding QP value for the intended video rate is fundamentally important in producing a correct predetermined video rate. In the SVBR algorithm, the video rate estimation/prediction is done at the rate controller engine, then the corresponding estimated QP value is fed into the real video encoder to generate the real encoded video rate. In the SVBR algorithm, the estimation/prediction of the intended data is moderately good. However, the problem is in generating the corresponding QP value, especially when the next video rate changes significantly. In this case, the estimated QP value will generate a significantly different video rate. Figure 5.2 illustrates this situation.

#### **5.2.1.2 Obtaining Correct Corresponding QP Value**

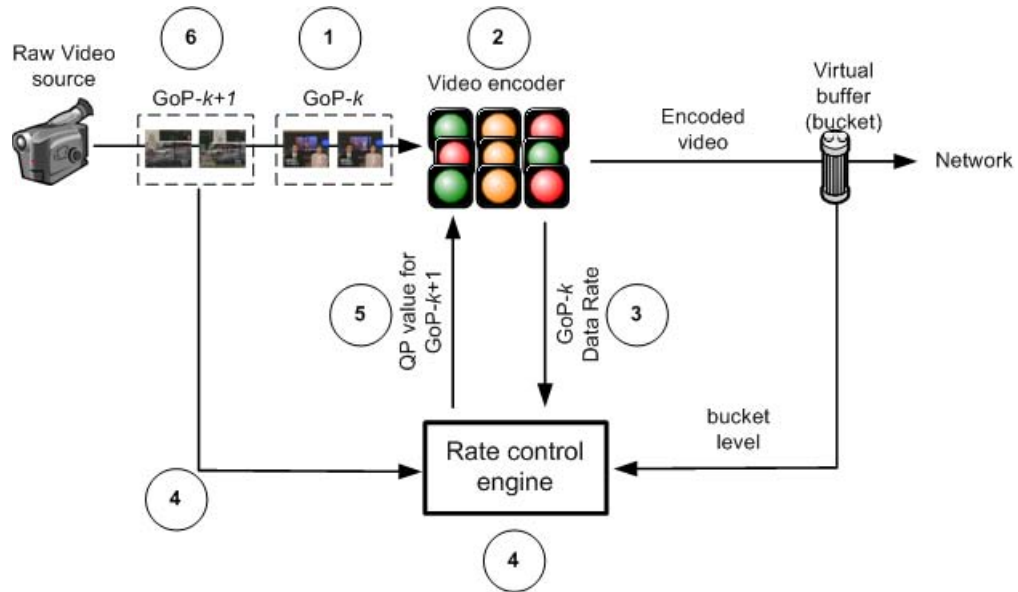
In a slight delay video application, while processing the GoP- $k$  video sequence, the rate control engine can obtain the GoP- $k+1$  information as demonstrated in Figure 5.3. From the figure, at step 4, before generate the QP value for GoP- $k+1$ , the application



**Figure 5.2:** Problem of generating the correct corresponding QP value



can process the GoP- $k+1$  video sequence. By acquiring certain GoP video sequence, a program can process that video part, such as to produce a list of corresponding QP-to-video rate. Table 5.1 displays a sample of two lists, of a low and an active video rate. An active video sequence usually holds a high QP-to-video rate for the first several QP values compared to a low active video sequence.



**Figure 5.3:** Getting information of GoP- $k+1$  video sequence while processing suitable QP value for GoP= $k+1$

**Table 5.1:** List of corresponding QP-to-video rate sample

QP for a low video rate	video rate (Bytes/Gop)	QP for an active video rate	video rate (Bytes/Gop)
2	78624	2	228816
3	52416	3	130032
4	40320	4	87696
5	35280	5	61488
6	33264	6	47376
:		:	
:		:	
30	16128	30	15120
31	16128	31	15120

By obtaining the list of corresponding QP-to-video rate, the rate controller does

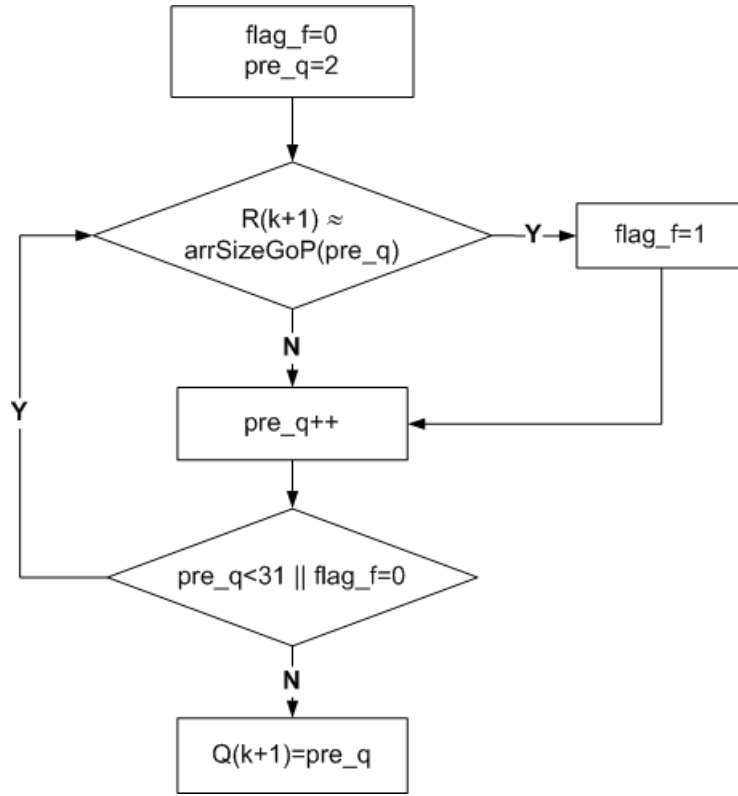
not only get a correct QP-to-video rate matching, but also a further opportunity to manipulate or shape the next video rate. For instance, the implication of using certain QP values for the next GoP video rate and for the bucket level, it can be assessed beforehand. Thus, the rate control engine can choose a more appropriate QP value.

In implementing the QP selection, when the list of corresponding QP-to-video rate is available, a simple data loop command is able to perform that task. The flowchart of getting the correct match of QP-to-video rate is shown in Figure 5.4. The following list is the description of the variables used in the flowchart;

- `flag_f`: a programming flag to indicate either the corresponding QP value for the intended video rate has been found or not. The 0 value means that it is not found yet.
- `pre_q`: a variable that contains a QP value.
- $R(k+1)$ : an intended video rate for GoP- $k+1$ .
- `arrSizeGoP()`: an array variable which contains a list of QP-to-video rate. The list of QP values is from 2 to 31. This list is used in the Evalvid-RA working environment (refer to [2]).
- $Q(k+1)$ : contains the selected QP value.

### 5.2.1.3 The Benefit Gained

Generally, by getting the correct matching between video rate and QP value, the shaping algorithm is more deterministic. The intended video rate can be obtained accurately. As described in the previous subsection, in the SVBR algorithm, the intended video rate is obtained by estimating the QP value for the next video rate. Then, the QP value is fed into the video encoder to compress the raw video. However, the changes in the next video sequence might be different from the previous ones,



**Figure 5.4:** A flowchart for a QP selection

which causes the generation of different video rates from the intended video rate. In some circumstances, the generated video rate might be extremely high/low.

In the new shaped algorithm, the next QP value is based on the list of the next GoP's corresponding QP-to-video rate. Thus, the chosen QP value closely produces the intended video rate. In the following discussion, some samples are highlighted to demonstrate how this mechanism is capable of achieving the intended design objective. Table 5.2 lists the outcome of QP-to-video rate for SVBR and SD-SVBR. Accordingly, Figure 5.5 displays the outcome of the corresponding video rate for SVBR and SD-SVBR.

From Table 5.2a, it can be observed that for GoP 201, the estimated video rate is 17714, but the obtained video rate is 6048. This usually happens when the VBR video rate changes its rate significantly from the previous rate. The estimated video rate is actually similar to GoP-200, which is using QP=7 to obtain 17136 Bytes/GoP.

However in GoP 201, the SVBR rate controller, although using QP=7, has obtained a big different value.

In SD-SVBR algorithm, as listed in Table 5.2b, the intended video rate at GoP 201 is 23160 Bytes/GoP. SD-SVBR chooses QP=2 and obtains 24192 Bytes/GoP. Although, it is not an accurate matching between the intended and the obtained video rate, this is the nearest video rate that can be gained from this GoP. The QP=3 for GoP 201 will produce 15120 Bytes/GoP, which is quite far from the intended video rate.

**Table 5.2:** QP-to-video rate for SVBR and SD-SVBR at GoPs 200-202

(a) QP-to-video rate for SVBR

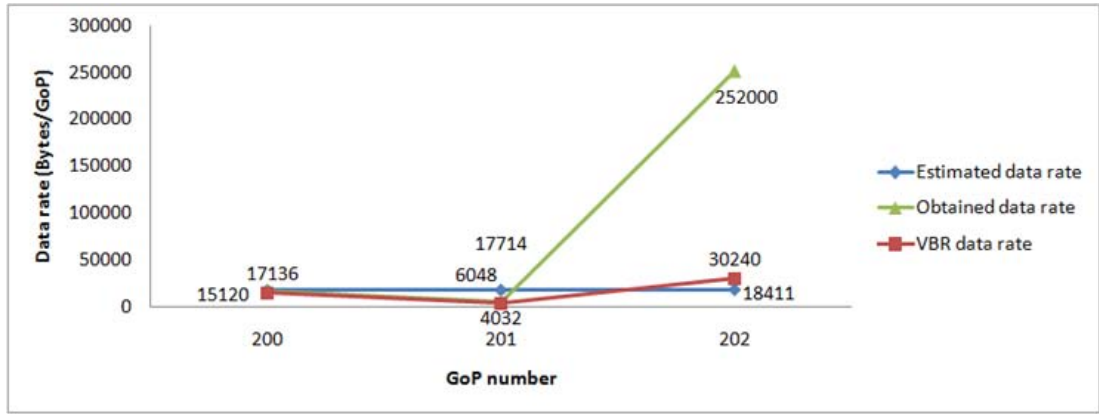
GoP #	Estimated video rate	QP	VBR video rate	Obtained video rate	Difference
200	17714	7	15120	17136	96.7%
201	17714	7	4032	6048	34.1%
202	18411	2	30240	252000	7.3%

(b) QP-to-video rate for SD-SVBR

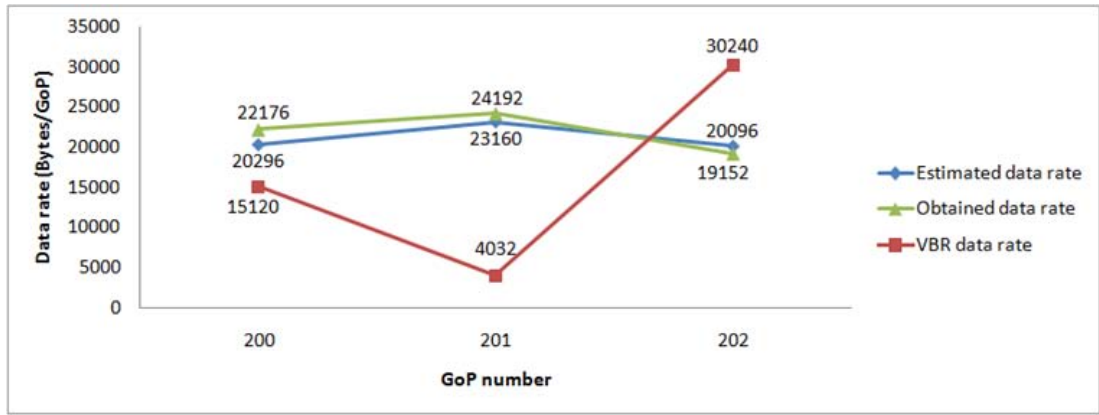
GoP #	Estimated video rate	QP	VBR video rate	Obtained video rate	Difference
200	20296	6	15120	22176	91.5%
201	23160	2	4032	24192	95.7%
202	20096	11	30240	19152	95.3%

In a similar trend, at GoP 202, the estimated video rate is 20096 Bytes/GoP, and SD-SVBR has chosen QP=11. This has generated the video rate for SD-SVBR as 19152 Bytes/GoP, and this is pretty close. In this context, the SD-SVBR algorithm has chosen the closest video rate available in the list of QP-to-video rate. Figure 5.5 presents these video rates in a line chart form.

As a result from this approach, it is clearly shown in Figure 5.5b that the video rates in SD-SVBR are adaptived to a moderately high video rate. The sharp decrease and increase, as shown in Figure 5.5a, are not sighted anymore, where previously, at one time a decrease to as low as 6048 Bytes/GoP was sighted and the next time it overly



(a) SVBR algorithm



(b) SD-SVBR algorithm

**Figure 5.5:** QP-to-video rate result for SVBR and SD-SVBR at GoP 200-202

burst to 252000 Bytes/GoP. SD-SVBR has been devised to be at a moderately high video rate, but not to the extent of bucket overflow.

The devised mechanism is working throughout all overflow or underflow phenomena, even in areas where there is no obvious sharp decrement followed by another sharp increment. For instance, the scenario at GoP 368-370, as shown in Table 5.3 and its corresponding line chart in Figure 5.6. It is obviously clear from Table 5.3a, that the differences between estimated video rates and obtained video rates in SVBR algorithm are quite big. It is more obvious at GoP 369, that the estimated video rate is 17378, whereas the obtained video rate is 59472. On the contrary to SD-SVBR algorithm, as shown in Table 5.3b, the differences between estimated video rates and obtained video rates are very small.

As a result, there are no sharp bursts occurring in SD-SVBR. As a conclusion, in SD-SVBR there will be no unwanted extreme decrease or increase in the video rate. By estimating correctly the intended video rate and its corresponding QP value, the SD-SVBR is able to choose the right QP value to produce the closest video rate for the estimated video rate.

**Table 5.3:** QP-to-video rate for SVBR and SD-SVBR at GoPs 368-370

(a) QP-to-video rate for SVBR

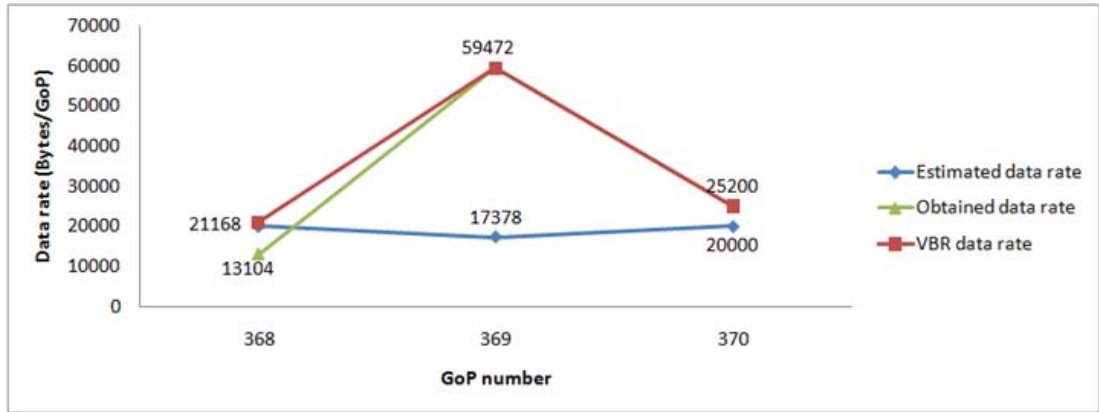
GoP #	Estimated video rate	QP	VBR video rate	Obtained video rate	Difference
368	20000	13	21168	13104	65.5%
369	17378	10	59472	59472	29.2%
370	20000	30	67536	25200	79.4%

(b) QP-to-video rate for SD-SVBR

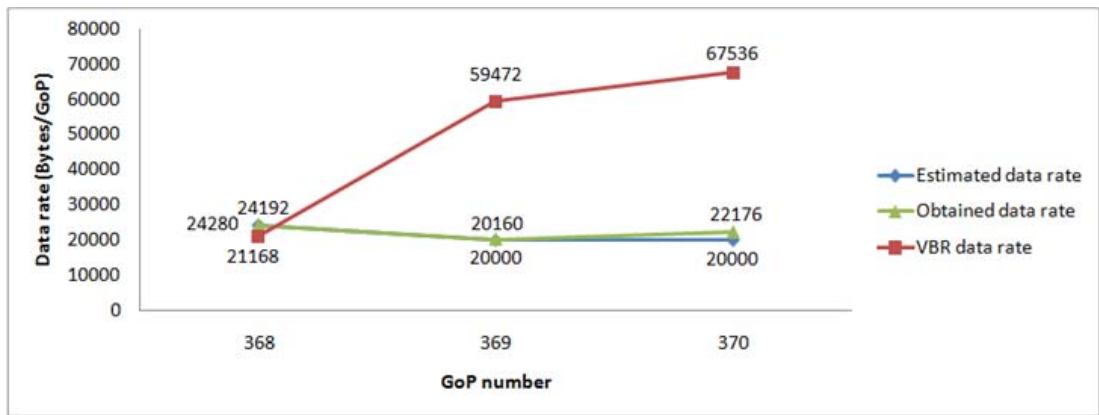
GoP #	Estimated video rate	QP	VBR video rate	Obtained video rate	Difference
368	24280	8	21168	24192	99.6%
369	20000	30	59472	20160	99.2%
370	20000	31	67536	22176	90.2%

### 5.2.2 Increasing the video rate

One of the principles of the SVBR algorithm is to let its video rate lie between the CBR and VBR video rates. Therefore, when both CBR and VBR are at the low rate, the SVBR video rate also tends to be at a low rate. By staying at the low rate for some time, the bucket level will be at the low rate as well.



(a) SVBR algorithm



(b) SD-SVBR algorithm

**Figure 5.6:** QP-to-video rate for SVBR and SD-SVBR at GoP 368-370

In order to increase the video rate, it cannot simply increase the rate. This is due to the fact that the leak rate is slower than the input rate. If the input rate is kept increasing, the bufer will easily overflow. It has to be controlled to the maximum of the leaky bucket size. In addition, it is not a good strategy to let the designated video rate fully occupy the bucket because of the following reasons:

- When the current video rate fully occupies the bucket, the next video rate has to be reduced tremendously, especially if the next video scene is in an active rate. This condition will create a significant negative fluctuation and definitely will make the viewing experience annoying.

- Even if some small space is available in the bucket, when the intended video rate is translated into QP value which is applied into the real video sequence, it might cause bucket overflow. This is because the QP value might not be able to get the exact intended rate and the closest rate is higher than the intended rate. In the case of an active sequence, the small changes in the QP value, might be translated to a big difference in the video rate.

#### **5.2.2.1 Design a Higher video rate Video Controller**

Since SVBR is dimensioned base on the leaky bucket algorithm, which has contributed substantially in terms of shaping the video rate within a permissible network bandwidth, the designation of a higher video rate should take bucket level utilization into account in the new algorithm. This is to prevent bucket overflow.

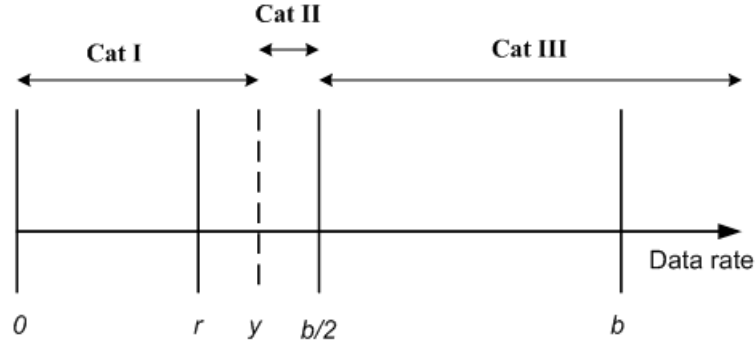
To increase the video rate, the intended video rate needs to be increased and the corresponding QP value should be decreased. However, as highlighted in the previous section, it cannot be simply increased in order to reduce the risk of bucket overflow and the fluctuation effect. Thus, an increment strategy based on scene activity video rate and the bucket level fullness has been devised. In addition, an appropriate space needs to be reserved in the bucket to reduce the video rate related negative fluctuation. Figure 5.7 shows some possible categorizations.

The VBR video rate in the range of 0 to  $r$  is a safe range to increase the SD-SVBR rate, even up to the size of the bucket. This is due to the fact that the range is lower than the leak rate and the incrementing up to the bucket size will not lead to bucket overflow. However, to increase the rate up to bucket size is also not a good strategy, since the video data is in variable bit rate, some spaces should be reserved for future increment.

Nevertheless, 25% of the bucket space is enough for several consecutive GoPs with more or less video rates. This can be translated into increasing the video rate, so that



the bucket fullness level is around 75%. This also can cover the video rate up to  $y$ , which is half of  $r$  and  $b/2$ . Thus, in Category I (Cat I) VBR video rate, the bucket fullness level is devised to be at 75% of bucket size. Higher than this level might risk the stability of the QP value, which is going to be elaborated in the next section. Lower than this, it might not create a good high video rate algorithm.



**Figure 5.7:** video rate categorization

The region after  $b/2$  in the figure is considered as an active sequence. This is because it is quick to fill up the bucket with a couple of consecutive GoP data transmissions. The 50% bucket utilization is considered good here. The area between  $y$  and  $b/2$  is also an active sequence, but with less degree compared to the region in category III. A 75% bucket utilization has been chosen for this category and is considered good. A 25% reserve space is still ample to accommodate that video rate. Two thirds of the bucket level is not selected here due to some forms of rate adaptations that will occur during the shaping processes, which will yield the video rate close to Category I.

In summary, the buffer fullness level is divided into three categories according to the VBR video rate, which are as follows;

- Category I: for the VBR video rate between 0 to  $y$ , the bucket fullness level is targeted around 75% (3/4 bucket level).
- Category II: for the VBR video rate between  $y$  and  $b/2$ , the bucket fullness level

is targeted around 75% ( $3/4$  bucket level).

- Category III: for the VBR video rate above  $b/2$ , the bucket fullness level is targeted around 50% ( $1/2$  bucket level).

#### 5.2.2.2 The High video rate Algorithm

The sample of C++ codes algorithm for increasing the video rate is shown in Figure 5.8. The following is the description of the algorithm:

```
1.  float paras = 0.75;
2.  int active = 0;
3.  if ( $r\_open < (r\_ + ((b\_/2 - r\_)/2))$ ) {
4.      paras=0.75;
5.      active=1;
6.  }
7.  else {
8.      if ( $r\_open < b\_/2$ ) {
9.          paras=0.75;
10.         active=2;
11.     }
12.     else {
13.         paras=0.5;
14.         active=3;
15.     }
16. }

17. If ( $lb\_X > r\_$ )
18.     Rtemp =  $paras * b\_ - (lb\_X - r\_)$ ;
19. else
20.     Rtemp =  $paras * b\_;$ 

21. If ( $Rtemp < r\_$ ) Rtemp= $r\_;$ 
```

**Figure 5.8:** Sample C++ codes for the increased video rate algorithm

- Line 1: “paras” is a variable to store the percentage of the buffer fullness level. The initial buffer fullness level is set at 75%.
- Line 2: variable “active” is used to keep the category of the video rate.

- Line 3-5: “ $r_{open}$ ” is a variable for VBR video rate, “ $b_{-}$ ” is the bucket size and “ $r_{-}$ ” is the leak rate. The first test here is to seek the range between 0 and  $y$  (refer to Figure 5.7). If this range is found then  $paras$  is set to 75% of the bucket and  $active$  is set as the first category.
- Line 8-10: Tests the range between  $y$  and  $b/2$ , and if it is found then  $paras$  is set to 75% of the bucket level and  $active$  is set to Category II (Cat II).
- Line 12-14: The “else” is to test whether the range is above  $b/2$ . If this is true, then  $paras$  is set to 50% of the bucket level and  $active$  is set as the third category.
- Line 17: Here, “ $lb_X$ ” is the bucket fullness level. If the bucket level is above  $r$ , then a targeted video rate is reduced further.
- Line 18: The “ $Rtemp$ ” is the targeted video rate. As explained in Line 17, “ $Rtemp$ ” has to be reduced to  $(X - r)$ , which is less than the targeted video rate. Here  $(paras * b_{-})$  produces the targeted percentage of bucket level size. This expression will avoid bucket overflow or become very close to the full level.
- Line 19-20: If  $X$  is less than  $r$  (the bucket fullness level is lower than the leak rate), the targeted video rate is set as the percentage of bucket level size. The extra increase in the targeted video rate is not needed here since it has been increased to the percentage level.
- Line 21: If the targeted video rate obtained is below the leak rate, the targeted video rate is set to the leak rate. This is to ensure that the final video rate is not at a low level, below the leak rate.

In addition to the implementation of the targeted bit rates according to the VBR video rate categorizations, which determines the activity in the video sequence, this algorithm also includes some controls to avoid bucket overflow and bucket under

utilization. This algorithm is also designed in such a way that the video rate is prone to be at a higher rate (refer to Line 18 and 20 in Figure 5.8). Here, the bucket utilization is maintained at 75% of the bucket size, thus, it avoids the previous SVBR limitation which tends to be at a low rate or lying between CBR and VBR video rates.

In maintaining the bucket utilization at a higher video rate, this algorithm intentionally reserves some spaces for video rate increment or decrement. This is important for the purpose of maintaining the stability of QP values, which will contribute to better user viewing experiences. The designation of QP value stability will be elaborated on in the next subsection.

### **5.2.2.3 High video rate Algorithm Achievement**

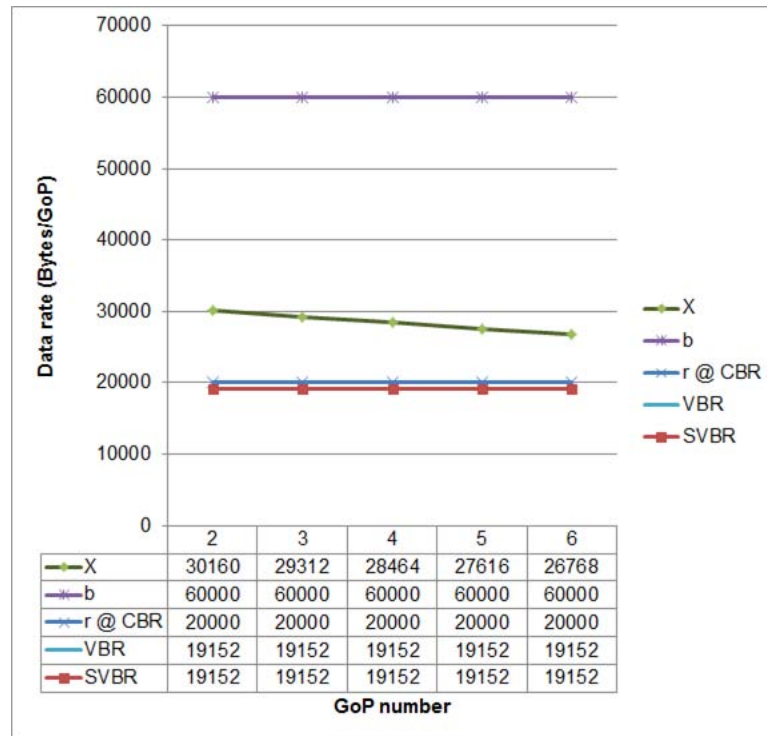
In SVBR, where both CBR and VBR are at the low rate, its video rate tends to be at a low rate as well. In SD-SVBR, since its video rate is not dimensioned based on CBR and VBR, regarding the bucket fullness level, its video rate is not limited to be between CBR and VBR.

As shown in Figure 5.9, when the CBR and VBR are at the 20000 and 19152 Bytes/GoP respectively, the SVBR is limited to the range, in which it gains 19152 Bytes/GoP. Even though the bucket fullness levels are quite high, SVBR does not really dimension its video rate to the bucket fullness level. The bucket level is used mainly to estimate its video rate either to be near the CBR or VBR. The other function of the leaky bucket is to control so that it does not overflow, but it is not really successful as described previously. The quite high bucket fullness levels, in this case, are due to the initial bucket setting and not the SVBR video rate bucket utilization.

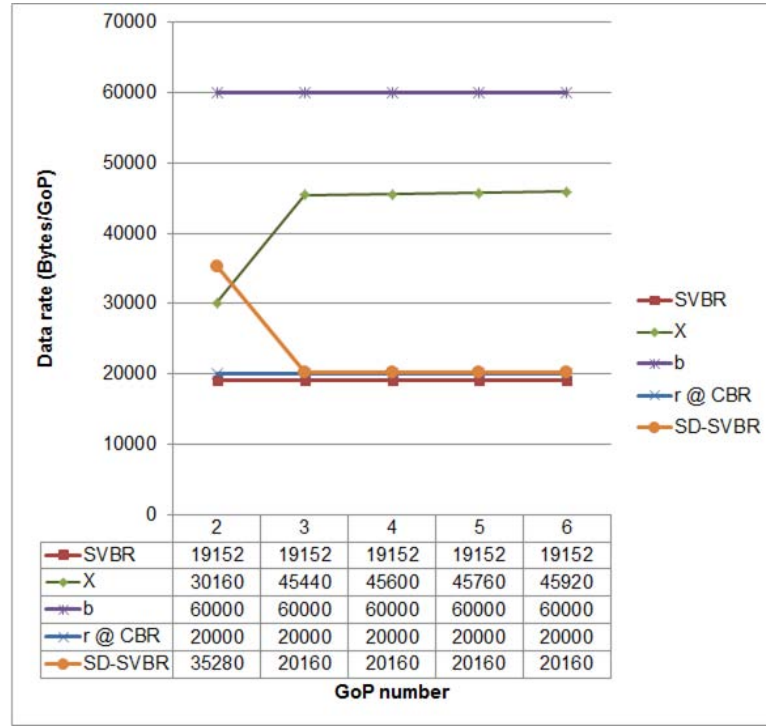
The high video rate principle in SD-SVBR can be observed in Figure 5.10. Besides lying between CBR and VBR, SD-SVBR gained a video rate higher than both; at GoP 2 it gained 35280 Bytes/GoP and 20160 Bytes/GoP for the rest. It got 35280 Bytes/GoP at GoP 2, as the algorithm wants the bucket fullness level to be at around

45000 Bytes (75% of the bucket size). The bucket fullness level for GoP 2 (after transmitting GoP 2 video sequence) can be seen at GoP 3, which is 45440 Bytes.

For the rest of GoP video rates, SD-SVBR gained 20160 Bytes/GoP given that there is no more space (for higher than these video rates) due to the 75% bucket fullness level constraint. However, these are still higher than the SVBR video rates and higher bucket utilization. From this data, it can be somewhat concluded that the target bucket utilization below 75% might not produce a high enough video sequence video rate. However, to increase bucket utilization more than this level might risk the instability of the QP value, which is going to be elaborated on in the next subsection.



**Figure 5.9:** SVBR at GoPs 2-6: Low rate trend



**Figure 5.10: SD-SVBR at GoPs 2-6: Breaking the SVBR limitation**

### Calculating a High video rate for SD-SVBR

The equations for Line 16-20 in Figure 5.8 can be written as follows:

if  $X(k+1) > r$  then

$$R_{temp}(k) = paras \times b - (X(k+1) - r) \quad (5.1)$$

else

$$R_{temp}(k) = paras \times b \quad (5.2)$$

Therefore, in increasing its video rate from GoP 2, it firstly compares  $X(k+1)$  and  $r$ . In calculating GoP 2 for SD-SVBR video rate,  $X(k+1)$  is equal to 30160 and  $r$  is equal to 20000. Another important information that should be obtained is the value of  $paras$ . Since the VBR video rate for GoP 2 is equal to 19152, the value for  $paras$  is set to 0.75 (refer to Line 3 of Figure 5.8).

Since  $X(k+1) > r$  then Equation 5.1 is employed,

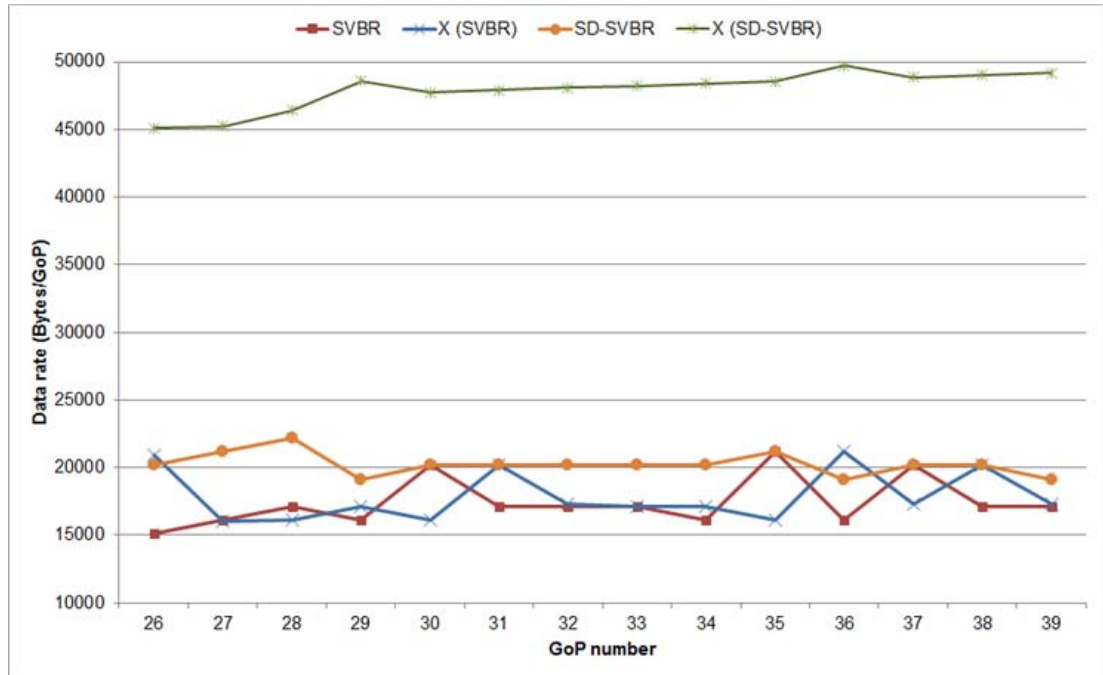
$$R_{temp}(2) = paras \times b - (X(k+1) - r)$$

$$R_{temp}(2) = (0.75 \times 60000) - (30160 - 20000) = 45000 - 10160 = 34840$$

In obtaining the QP for 34840, SD-SVBR uses QP=4, which generates the video rate equal to 35280 Bytes/GoP. This is a much higher video rate compared to 19152 Bytes/GoP gained by SVBR algorithm.

### SD-SVBR at a Low SVBR Bucket Utilization

Figure 5.11 shows another result of the high video rate algorithm. This is the case where both of the SVBR video rate and its respective bucket utilization are at a low rate. Its video rate is around 17400 Bytes/GoP and its respective bucket utilization is around 17800 Bytes/GoP. By securing the bucket utilization in SD-SVBR to around 45000 Bytes/GoP, its video rate is higher at around 20200 Bytes/GoP. Thus, it can be concluded that the high video rate is more beneficial where both of CBR and VBR are at low rates.



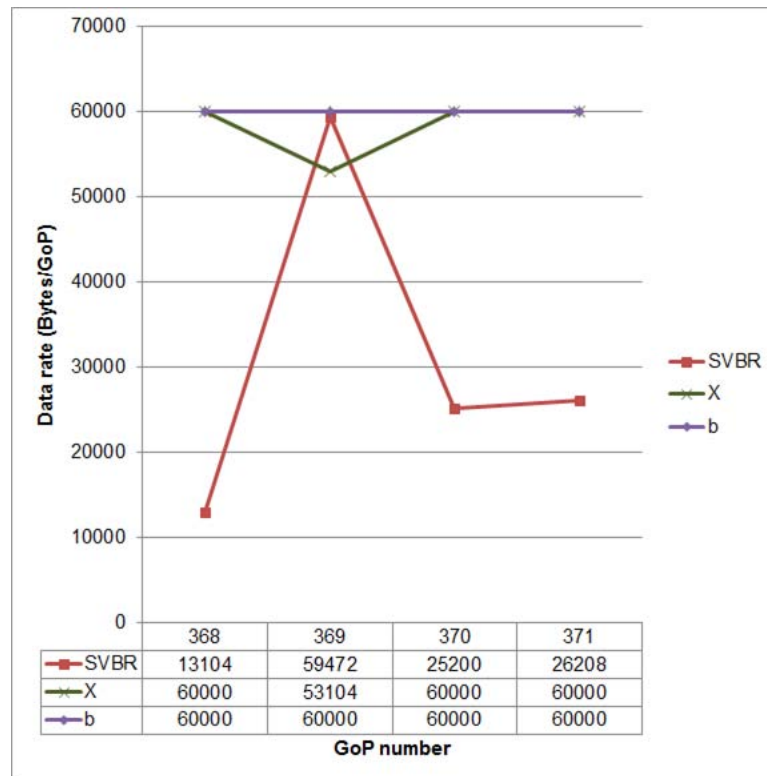
**Figure 5.11:** SD-SVBR at low SVBR bucket utilization

### SD-SVBR at a High video rate Video Sequence

It is very important to control the input transmission rate in order to avoid data overflow. This is especially true where the video sequence is in a highly active mode. For instance, in reference to the transmission rate at GoP 369 in Figure 5.12, the SVBR video rate is at 59472 Bytes/GoP and it is below the bucket size limit. Even though the bucket fullness level is stated as 60000 Bytes after the GoP 369 transmission, in the real transmission it has very much overflowed. Before the GoP 369 transmission, the bucket fullness level is at 53104 Bytes. By taking account that the leak rate is 20000 Bytes/GoP and the real transmission is 59472 Bytes/GoP, the bucket fullness level will be at 92576 Bytes. This is 32576 Bytes overflow. The calculations are as follow:

$$53104 + (59462 - 20000) = 92576$$

$$92576 - 60000 = 32576$$

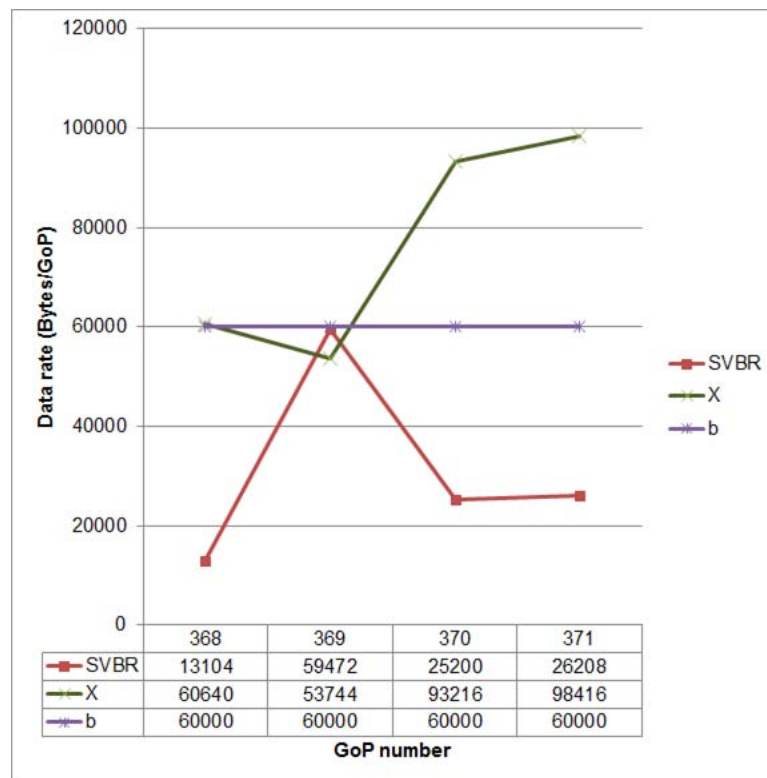


**Figure 5.12:** SVBR at GoPs 368-371: Overflow bucket trend

For GoP 370, SVBR transmits another 25200 Bytes/GoP while its leak rate is



20000 Bytes/GoP, which means it adds an additional 5200 Bytes to the already overflowed bucket. Now, the bucket virtually contains 97776 Bytes. The transmission after that also adds another over-supply video rate to the bucket. This scenario will be ongoing if this pattern is continuing. Thus, the real overflow in the SVBR algorithm can be displayed as in Figure 5.13. Actually the small overflow has occurred at GoP 368, where the bucket fullness level was at 60640 Bytes (refer to Subsection 6.1.3.1). The overflow at GoP 370 and 371 are very high.



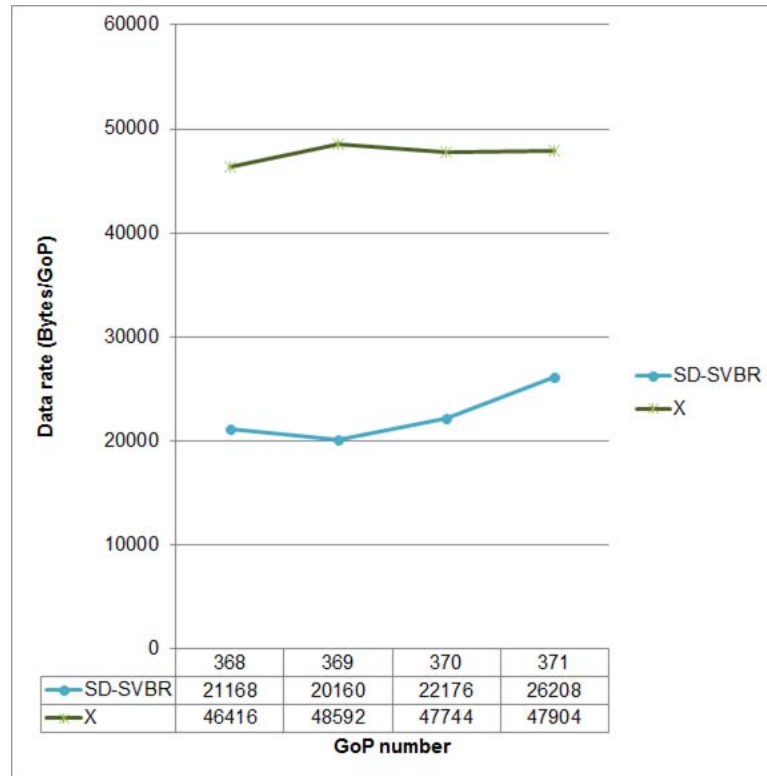
**Figure 5.13:** SVBR at GoPs 368-371: The real overflow

The bucket overflow does not happen in SD-SVBR because the algorithm is based on bucket level. In addition to that, it is controlled so that any intended video rate should be produced at less than the bucket size. This principle is controlled by Equation 5.1. For example, in producing the video rate for GoP 369, SD-SVBR performs the following principles:

- The intended rate is set to 50% of the bucket level or equal to 30000 Bytes.

Although this is not a high enough rate, in an active scene especially in an area near to or above the bucket size, any changes in QP value will produce a big difference in the video rate.

- It is not always able to gain a high video rate with a targeted bucket utilization at 50%, especially if the previous GoP bucket utilization has highly utilized the bucket. This is very true when the case of Equation 5.1 is applied.
- Thus, any increment to the video rate still can be shaped within a permissible level.



**Figure 5.14:** SD-SVBR at GoPs 368-371: Maintaining the bucket level below  $b$

### 5.2.3 Smoothing the Fluctuation

It is not always possible to have a constant QP value due to the variable bit rate generated from the activity of a video sequence. As with the problem encountered

in VBR, sometimes the activity of a video sequence generates overflow data. Thus, the overly high or low activities in the video sequence need to be controlled. This is why the algorithm is called the shaped VBR, where to a certain extent, the variabilities of the video rate have to be controlled, thereby to avoid over burstiness.

One of the approaches in order to gain a better constant QP value is to follow the VBR video rate wherever possible, but to shape the rate when there is an occurrence of over burstiness. Although it seems ideal, this is not a good strategy. In this strategy, the rate controller might get constant QP values to a certain extent, but:

- it might be prone to getting a low video rate, if the VBR is at a low rate; and
- it might end up with many fluctuation events. When the bucket is full or almost full, by following the VBR video rate, the rate controller engine is forced to drastically reduce the video rate. This creates an obvious change in the video rate (and to the effect of viewing smoothness as well). In the end, it might create a fluctuated QP value.

However, in shaping a VBR video sequence by employing the leaky-bucket algorithm, like in the SVBR, the negative fluctuations are quite obvious as elaborated in the previous chapter. This occurs especially where the gap between  $r$  and VBR is quite wide. Thus, a good algorithm should have a good balance between a high video rate and a constant QP value. Besides that, the bucket utilization should be optimum, and not under utilized or overflowed.

Since the shaped VBR is based on the leaky bucket algorithm, the reserve space in the bucket should be taken into account. The strategy is to reserve some space in the bucket, so that, when the algorithm wants to follow the previous QP value, which might add some data in the bucket, it can be allowed. However, if the space is very small as a result of accommodating previous video rate video sequence, the next video rate should be reduced sharply since the space is limited now. This, again, creates an

obvious fluctuation. If the reserve space is big, although it prepares ample space for a future drastic change to a high video rate, it means the algorithm might be settled at a very low rate, especially when the low rate pattern is continuing for a long period.

Thus, to reserve enough space in order to enable the algorithm to follow the previous QP value, the strategy based on the activity of a video sequence, as adopted in the previous algorithm, is followed. Then, the previous QP values will be followed until the reserve space is not more  $\frac{5}{6}$  of the bucket size. This is to allow for “softer” fluctuation as in the case where the video sequence activity change.

### **5.2.3.1 The Smoothing QP Value Principle**

In smoothing the QP value, the decision to be made is either to follow the previous QP value or not. The new QP value resulting from the intended video rate might be the same, being one or two units different from the previous QP value, or the difference is quite big. Another smoothing possibility is to make some adjustment in the case of the QP value differing largely from the previous one. In deciding which QP value to be used, several factors need to be considered:

- How many QP values can be followed?
- What is the effect to the bucket level?
- What is the effect of following QP values on the video rate, in several other GoPs, etc.?

### **The Effect of Following the Previous QP Value**

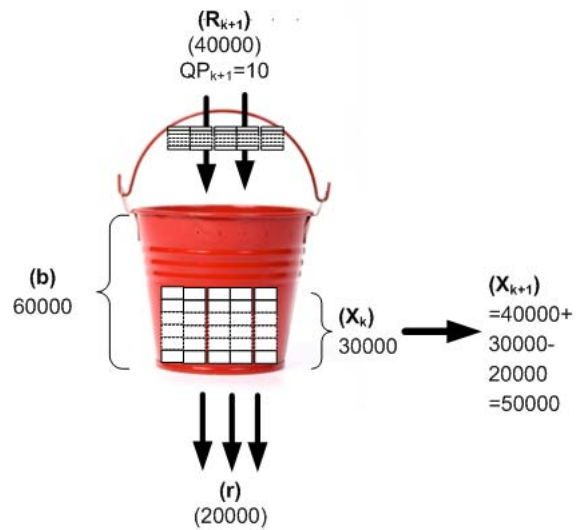
By following the previous QP value, the effect of it should be examined. For instance, say that the bucket fullness level is 30000 Bytes, the intended video rate for the next GoP is 40000 Bytes/GoP and the corresponding QP value for it is 10; by using this intended QP with its corresponding video rate, the bucket fullness level will become

50000 Bytes or 5/6 of the bucket size. Say that the previous QP value is 6. By using the new QP value which is 6, instead of 10 to produce the video rate around 40000, the QP=6 might be producing a very high video rate. Say that the new QP value will generate the video rate around 60000 Bytes/GoP. Thus, instead of producing 40000 Bytes/GoP, now the algorithm produces 60000 Bytes/GoP which is overflowed by as much as 10000 Bytes. Figure 5.15 illustrates this scenario.

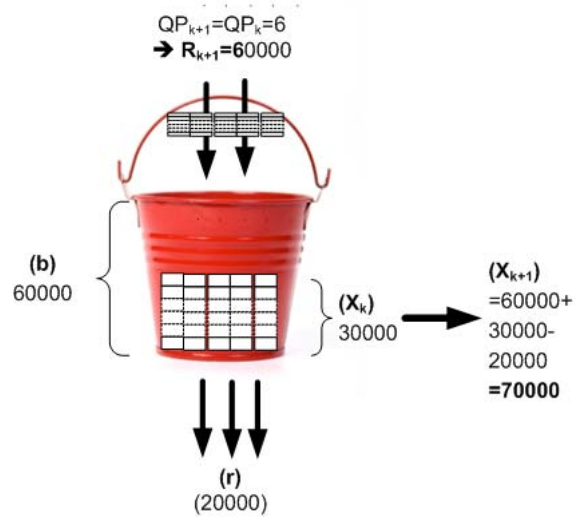
The reverse might be occurring in the case when the previous QP value is bigger than the intended QP value. For instance, say that the bucket level is 19000 Bytes, the intended video rate for the next GoP is 30000 Bytes/GoP, and the corresponding QP value is 8. By using this intended QP value with its equivalent video rate, the bucket fullness level will become 29000 Bytes or almost 50% of the bucket size. Say that the previous QP value is 12. By following the new QP value, it generates a lower video rate, say 16000 Bytes/GoP. This low video rate will lead to a lower bucket fullness level, in this case the bucket fullness level is down to 15000 Bytes. Consequently, this creates an under utilization of bucket space.

The following general observations have been used in this smoothing principle;

- One or two value difference in the QP value for Category II and III still can be followed.
- Up to four units difference in the QP value for Category I can be followed.
- For a large difference in the QP value, a middle value should be used.

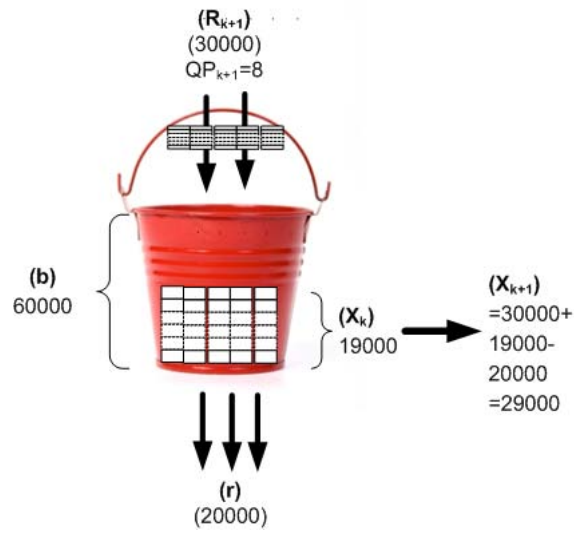


(a) Using the intended QP value

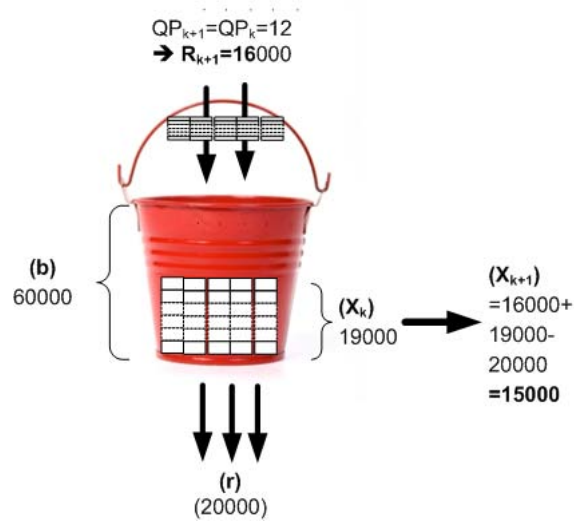


(b) Using the previous QP value

**Figure 5.15:** The effect of following a smaller QP value



(a) Using the intended QP value



(b) Using the previous QP value

**Figure 5.16:** The effect of following a bigger QP value

### The Smoothing Principle and Strategy

Therefore, the following steps have been devised for achieving a high video rate with smoother QP values:

- i. Estimating an intended video rate which can produce a suitable high video rate.

This is the same approach as explained in Subsection 5.2.2.

- ii. If the intended video rate produces a different QP value than the previous QP value, determine whether it can use the previous QP value or the new QP value.

Generally, it can use the previous QP value if the following conditions prevail:

- a. still at a high video rate,
- b. does not overflow,
- c. does not under utilize the bucket, and
- d. still reserves some more space for future adjustment to ensure that the three above conditions can still be met if step 2 occurs again.

- iii. The following are other assumptions, considerations, and steps employed in the SD-SVBR algorithm for the purpose of getting smoothed QP values:

- a. Usually a difference in 1 or 2 in the QP value will not create an adverse impact if good policing measures are implemented. These policing measures should consider the bucket reservation space, the current video rate, and the impact to the bucket level fullness when using a dissimilar QP value.
- b. In determining either to follow the previous QP value or not; in the case where the previous QP value is lower than the intended QP value, which might risk bucket overflow, the following measure has been enforced; in the case that the difference is 1 or 2 units for the video rate in Category II (moderately active video sequence), and the video rate in Category III (an active video sequence), the bucket fullness level, after following the previous QP value, is less than  $\frac{5}{6}$  of the bucket size. The strategy here is to let the bucket level reserve at least  $\frac{1}{6}$  space in the bucket after following the previous QP. By reserving  $\frac{1}{6}$  space in the bucket, for the next QP value, the algorithm does not have to choose a very high QP value due to



the limited space available in the bucket, and the video rate does not have to be reduced sharply.

- c. For the video rate in Category I, small changes in QP value usually give less impact to the bucket utilization. Thus, a difference of up to 4 QP value points is still considered acceptable. However, the measure that needs to be put in place is for control, so that when the trend is continuing, it does not come to overflow. The same strategy as in Category II and III is used, which is to let the bucket level reserve at least  $1/6$  space in the bucket after following the previous QP.
- d. In the case where the previous QP value is greater than the intended QP value, there will be no issue of bucket overflow. This is because, a greater QP value means a lower video rate. The lower video rate will not cause bucket overflow. However, in the case of active video rate as in Category II or III, the risk of bucket underflow is real, even the difference in QP value is just 1 or 2. Thus, in order to control the bucket from the under utilization condition, the bucket fullness level, after following the previous Q, should not be less than half of the bucket level. This is to ensure that the video rate is kept at quite a high rate.
- e. For the video rate in Category I, the difference of up to 4 QP value points is still considered acceptable. Again, the measure that needs to be put in place is for control, so that when the trend is continuing, it does not underflow. This will defeat the purpose of getting a constant QP value because the algorithm gains a constant QP value with a low video rate. The measure in place is to allow the use of the previous QP value so long as the video rate is still above the leak rate, which is more than half of the bucket level. By implementing in this way, the algorithm will not tend to be at a low rate, because it has started with a high rate and the bucket utilization will not fall

below half the bucket size.

- iv. It is difficult to create a constant QP value by following the previous QP value if the difference is bigger than 4. This is because it might quickly create an overflow or underflow phenomena, or an instance of almost full or almost empty bucket level. Thus, to reduce the big difference between an intended QP and the previous QP value, SD-SVBR employs a smoother approach if conditions allow. Instead of letting the next GoP use the intended QP value for the next QP, but at the same time it cannot follow the previous QP value since the difference is big, SD-SVBR chooses a middle value between them. This is only applicable for the video rate in Category I with the next bucket fullness level being bigger than  $2/3$  the bucket size.

#### 5.2.3.2 Smoothing QP Value Algorithm

Figure 5.17 shows the sample of C++ code employed in SD-SVBR algorithm in order to make the QP value change in a smoother way, as compared to the SVBR algorithm. The following is the descriptions and explanations of the codes:

- Line 1: The “next\_X” stores the bucket fullness level after submitting the current GoP data if it uses the previous QP value (this previous QP value will just be written as  $Q$  after this).
- Line 2: The “next\_Xpreq” stores the bucket fullness level after submitting the current GoP data if it uses the new estimated QP value (this new estimated QP value will be called  $pre_q$  after this).
- Line 3: Checks whether  $pre_q$  is 1 or 2 units after  $Q$  ( $pre_q$  is 1 or 2 units bigger than  $Q$ ). This means that if  $pre_q$  uses  $Q$  as the next QP value, it decreases the number of the estimated QP value. The implication of this implementation is

```

1)  int next_X = (saiz_gop[Q]-r_)+lb_X;
2)  int next_Xpreq = (saiz_gop[pre_q]-r_)+lb_X;
3)  if (pre_q==Q+1 || pre_q==Q+2) {
4)      if ((active==2) || (active==3))
5)          if (next_X <= (0.83333*b_)) {
6)              pre_q = Q;
7)          }
8)  }

9)  if (pre_q==Q-1 || pre_q==Q-2) {
10)     if ((active==2) || (active==3))
11)         if (next_X >= (b_/2))
12)             pre_q = Q;
13) }

14) if (Q-pre_q < 5 && Q-pre_q > 0) {
15)     if ((active==1) && (next_X >= (b_/2)) && (next_X!=next_Xpreq))
16)         pre_q = Q;
17) }

18) if (pre_q-Q < 5 && pre_q-Q > 0) {
19)     if ((active==1) && (next_X <= (0.83333*b_)))
20)         pre_q = Q;
21) }

22) if (Q-pre_q >= 5) {
23)     if ((active==1) && (next_Xpreq > (0.6667*b_))) {
24)         if ((saiz_gop[pre_q]-saiz_gop_Q) > (0.08333*b_)) {
25)             pre_q = ((Q-pre_q)/2)+1+pre_q;
26)         }
27)     }
28) }

```

**Figure 5.17:** Sample C++ codes for the QP smoothing algorithm

that it employs a higher video rate than the estimated one. This might cause an overflow if it is not controlled carefully.

- Line 4-5: In determining whether to follow the previous QP value for Category II and Category III, the bucket fullness level should be less than 5/6 the bucket size.
- Line 5-6: If the above conditions are met then the next QP value will use the previous QP value.
- Line 9: Check whether  $pre_q$  is 1 or 2 units before  $Q$  ( $pre_q$  is 1 or 2 units smaller than  $Q$ ). This means that if  $pre_q$  uses  $Q$  as the next QP value, it increases the estimated QP value. The implication of this implementation is that it employs

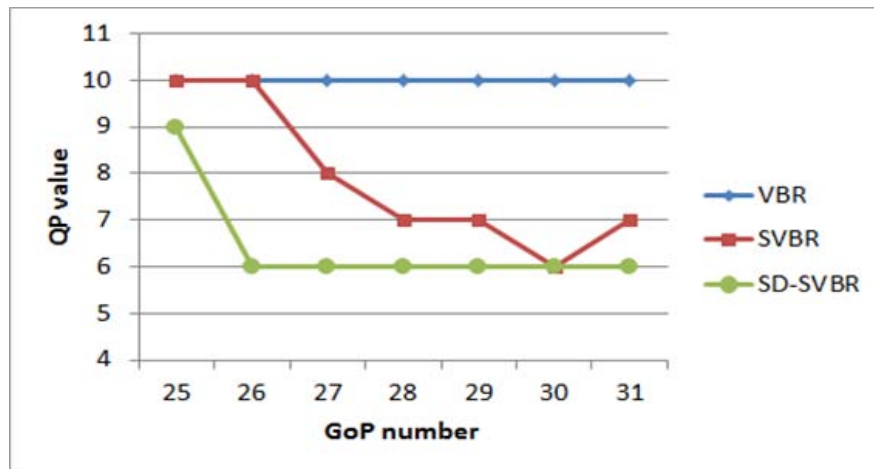
a lower video rate than the estimated one. The under utilization of the bucket might occur.

- Line 10-12: In order to avoid the situation of bucket under-utilization, for the video rate in Category II and Category III, the bucket fullness level after following the previous QP value should not be less than half of the bucket size. If the condition is met then the next QP value will use the previous QP value.
- Line 14: For the video rate in Category I, the small changes in QP value usually give less impact to the bucket utilization. Thus, the difference of up to 5 QP value points is still considered acceptable. Line 14 is devised to check whether  $pre_q$  is smaller than 5 units of the QP value.
- Line 15 and 16: The next QP uses  $Q$  value if the video rate is in Category I and the next bucket fullness level is above half of the bucket size. The “(next\_X!=next\_Xpreq)” test examines whether the bucket fullness level is similar when the next QP value uses the previous QP value. If they are same (next\_X=next\_Xpreq), then use a smaller QP value which is  $pre_q$  (do not execute command Line 16).
- Line 18-20: These commands are the reverse of Lines 14 to 16. The main measure here is to prevent the situation of bucket under-utilization after following the previous QP value. The measure here is that the bucket fullness level should not be less than half of the bucket size. If the condition is met then the next QP value will use the previous QP value.
- Line 22-25: In shaping the QP value according to the previous  $Q$  value, there will be cases that the changes in  $pre_q$  is more than 4 units. To reduce the wide difference between  $Q$  and  $pre_q$ , SD-SVBR employs a smoothing approach if conditions allow. This smoothing approach means to choose a middle value.

This only applicable for the video rate in Category I and the next bucket fullness level is higher than  $2/3$  of the bucket size. The other condition, in order to apply this smoothing approach, is that the difference in the next GoP size (between  $Q$  and  $pre_q$  values) is higher than  $0.5/6$  of the bucket size. Line 25 calculates the middle value for  $pre_q$ . If the original  $pre_q$  is 4 and  $Q$  is 10, then the final  $pre_q$  will be 8, and if the  $Q$  is 11, then the final  $pre_q$  will be 8 as well. The same value is obtained because of the round-up expression in Line 25.

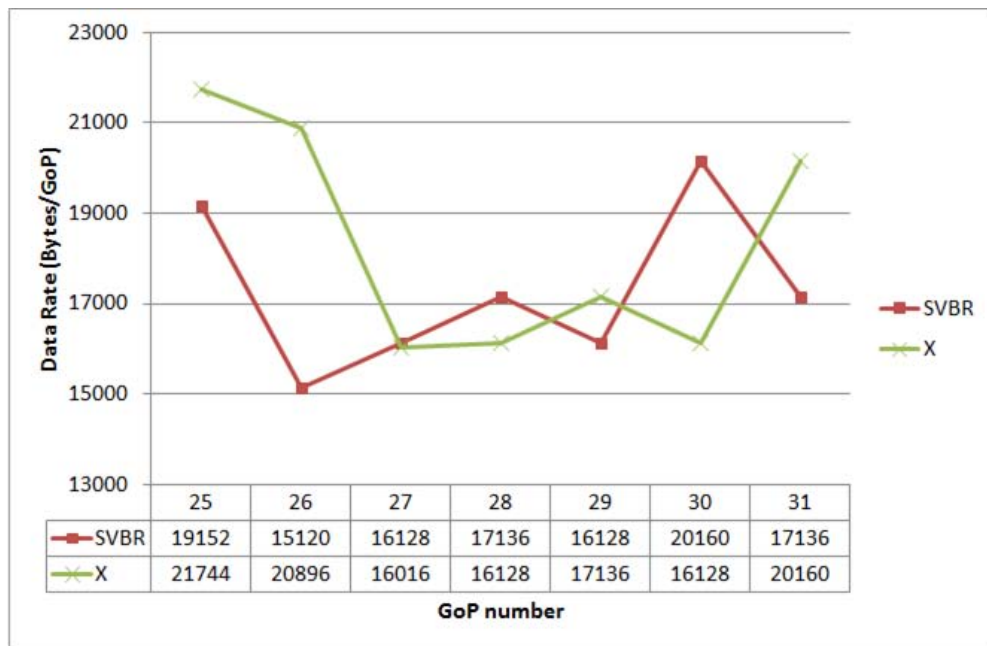
### 5.2.3.3 Effects of the Smoothing QP Value

The behaviour of SD-SVBR as compared to SVBR in smoothing the QP values can be examined in Figure 5.18. From this figure, it can be observed that in SVBR from GoP 26 to 30, the QP values are decreased from 10 to 6, whereas, SD-SVBR chooses to follow the previous QP.



**Figure 5.18:** Smoothing QP values at GoPs 25-31

Although, in SD-SVBR the GoP 27-31 follow the QP value for GoP 26, it does not mean that it tends to be at a lower rate than SVBR. The respective video rates gain by both algorithms are presented in Figure 5.19 and 5.20. These are due to the high video rate measures which have been implemented in the SD-SVBR algorithm.

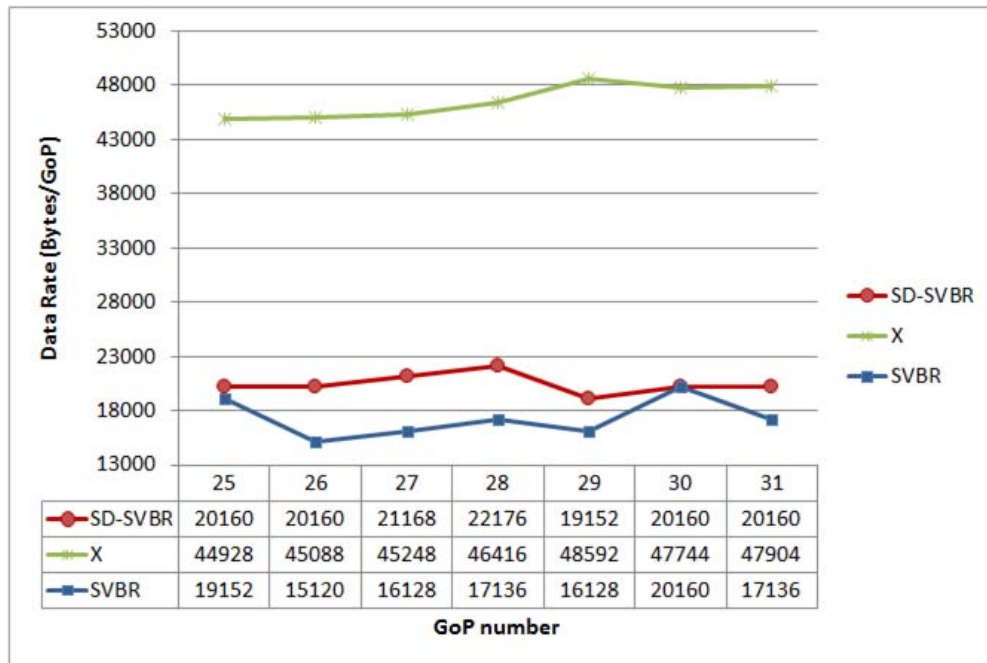


**Figure 5.19:** Corresponding video rate for SVBR algorithm at GoPs 25-31

Among the controls which have been implemented are to assign a high bucket utilization and to ensure that it does not degrade below the leak rate. However, a high bucket utilization should not be implemented as it generates bucket overflow or at an almost full bucket scenarios. It can be clearly observed that in the SD-SVBR algorithm, the bucket utilizations are around 45000 Bytes/GoP, whereas in SVBR, they are around just 20000 Bytes/GoP. Accordingly, the video rates in SD-SVBR are significantly at a higher rate than SVBR. However, the video rate for SD-SVBR at GoP 29 falls below 20000 Bytes/GoP, which is due to the implementation of the SD-SVBR program that uses the nearest corresponding video rate.

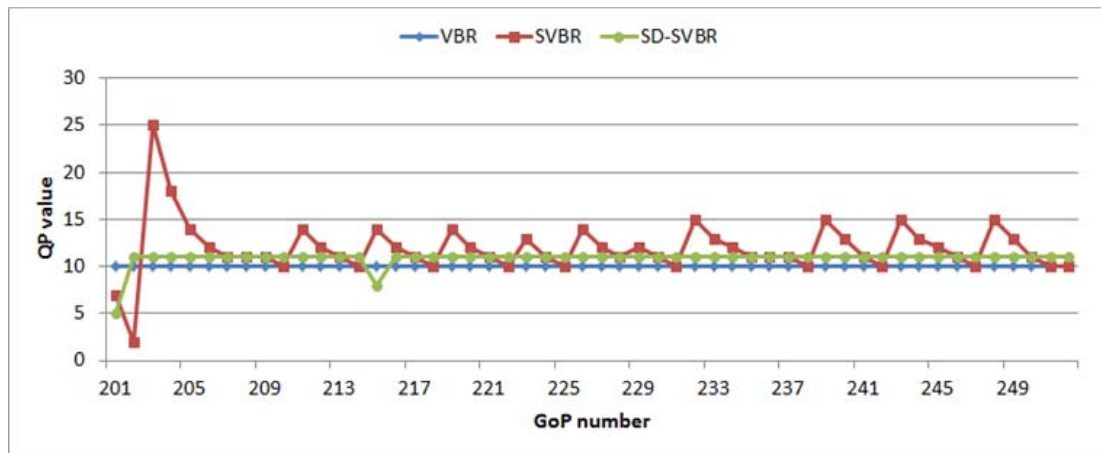
### **SD-SVBR is Able to Gain a Longer QP Constant**

By designing a smoothing algorithm with ample space in the bucket, a longer QP constant can be gained. This mechanism can be observed from GoP 201 to 253 in Figure 5.21. In that case, it can be clearly seen that the SVBR algorithm gains a high fluctuation in the QP values. On the other hand, after SD-SVBR uses QP=5 in GoP

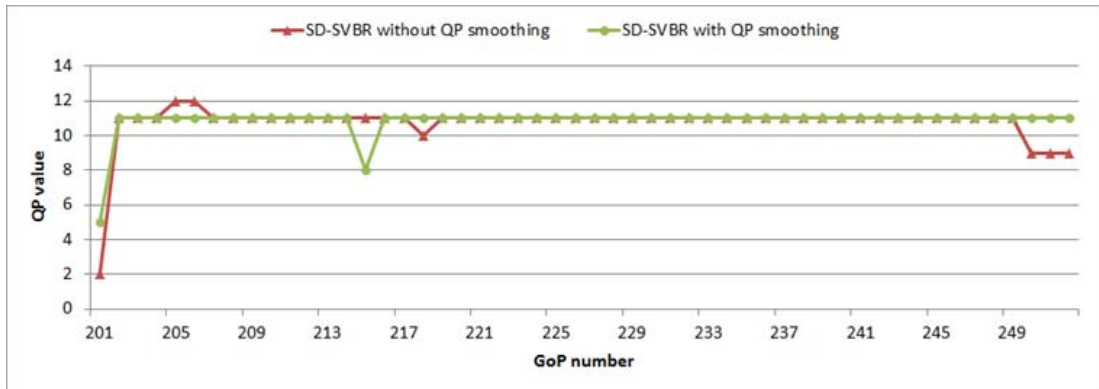


**Figure 5.20:** Corresponding video rate for SD-SVBR algorithm at GoPs 25-31

201, it chooses QP=11 and kept using that value. Only in GoP 215 does SD-SVBR choose to gain a higher video rate by using QP=8. Without a smoothing approach, SD-SVBR still gains considerable QP value stability, but it is not as good as SD-SVBR with a smoothing approach, as can be seen in Figure 5.22.



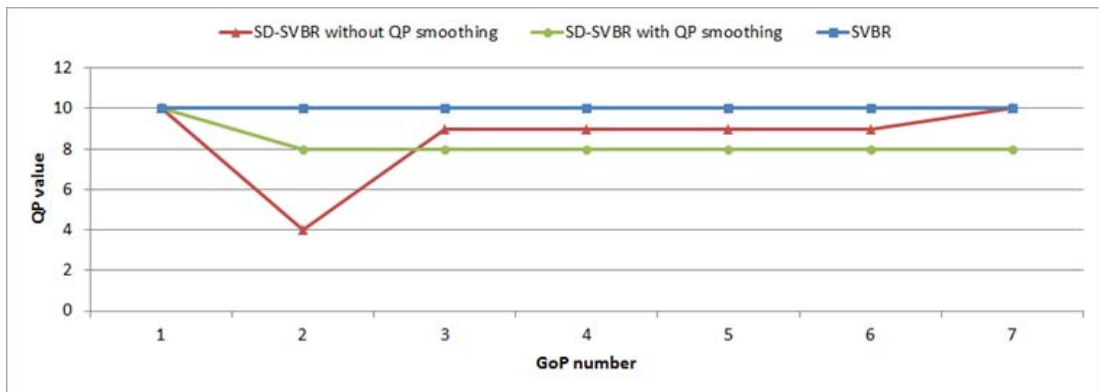
**Figure 5.21:** A longer SD-SVBR QP constant at GoP 201-253



**Figure 5.22:** Smoothing and non-smoothing SD-SVBR QP value for GoP 201-253

### The Smoothing Approach Effect

Without the smoothing approach, where the high difference between the intended QP and the previous QP is left as it is, there will be several events that can be significantly observed. For instance, it can be observed between GoP 1 and GoP 2, as shown in Figure 5.23. As illustrated in this figure, without the smoothing approach, the QP value from GoP 1 to 2 and the rest are changed from 10 to 4, then stable at 9, and in the end back to 10. With the smoothing approach, by taking a middle value, which is 8; the QP value changes from 10 to 8, then it is able to keep that value until GoP 7.



**Figure 5.23:** The effect of QP smoothing approach at Gop 1 and 2

Even though, in the above described scenario, the SVBR got the most stable QP value of 10, it is the highest QP value. This can be translated as the lowest video rate, which will be explained later. For the case of SD-SVBR without QP smoothing, it



started by getting the lowest QP value, which is a change from 10 to 4, but then its QP values increased back to 9, then ends with 10. Meanwhile in SD-SVBR with QP smoothing, it gets a stable 8 QP value, making it the most stable of the lowest QP values.

As described previously, the lowest-stable QP values will contribute to the highest video sequence video rates. As mentioned above, SVBR, which got the highest QP values, provides the smallest video sequence video rate. Figure 5.24 illustrated this scenario. The SD-SVBR without QP smoothing started by obtaining 35280 Bytes/GoP after getting 19152 Bytes/GoP, then only settling at 20160 Bytes/GoP for the rest of GoPs. In SD-SVBR with QP smoothing, after getting the initial 19152 Bytes/GoP on the first GoP, it then gains 21168 Bytes/GoP, followed by 22176 Bytes/GoP, and ends up with 26208 Bytes/GoP. Thus, it can be concluded that SD-SVBR with QP smoothing is able to gain the most stable QP values with the highest video rate.

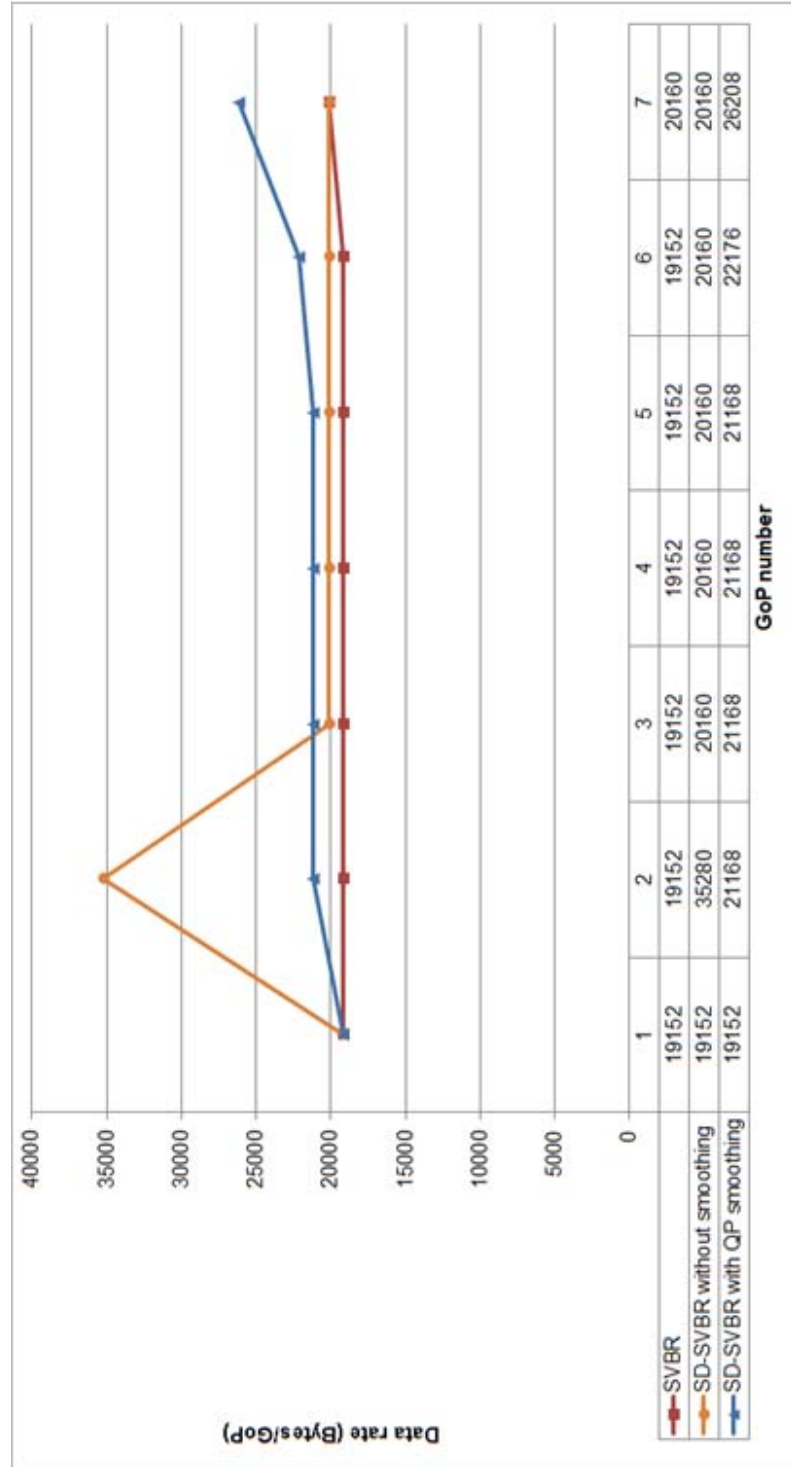
## **5.3 SD-SVBR Implementation**

As stated previously, the SD-SVBR algorithm has been implemented in the Evalvid-RA [168] environment. This environment is an extended environment of the Evalvid-ns2, which is another extended work of the original Evalvid, as discussed in Subsections 2.6.3 and 2.6.4.

### **5.3.1 Work Environment Architecture**

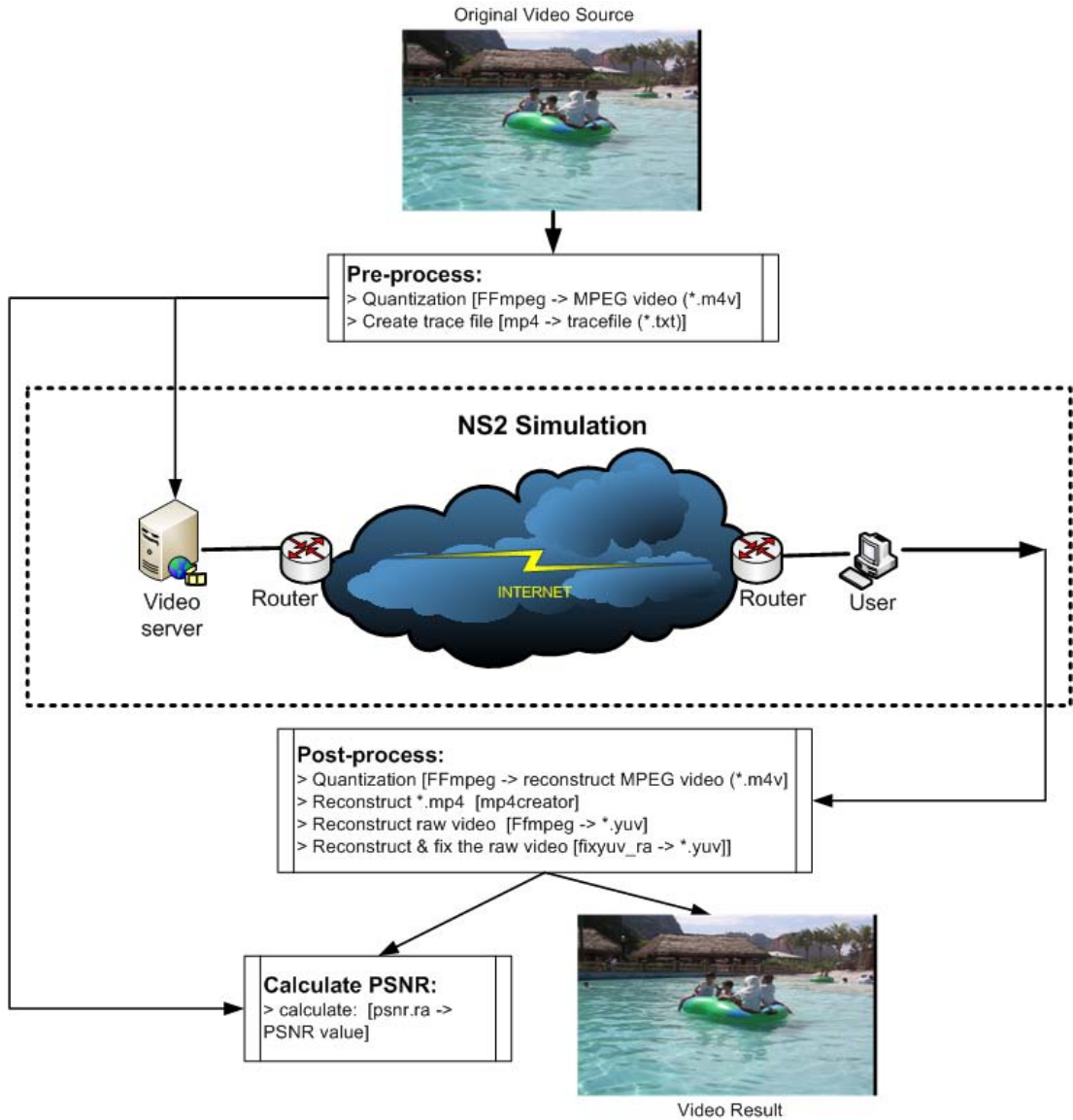
The EvalVid-ns2 and Evalvid-RA consist of three parts, which are:

- pre-process part or trace file generation processes,
- video transmission simulation part or the application/algorithm processes, and
- a post-process part or video performance evaluation processes.



**Figure 5.24:** The effect on the video rate of QP smoothing approach at GoP1 and 2

Figure 5.25 depicts the three parts of the Evalvid-RA environment.



**Figure 5.25:** Evalvid-RA environment parts

As compactly stated in [2], the key idea of the pre-process part is to encode the media with open loop VBR for all possible QP values, store the frame sizes per QP value in separate files, so that the online rate controller in the network simulator can select a new QP value and get the correct frame sizes from the corresponding trace file. For the detailed processes of the pre-process part, please refer to [2, 168].

In the video transmission simulation part, the processes that are involved include:

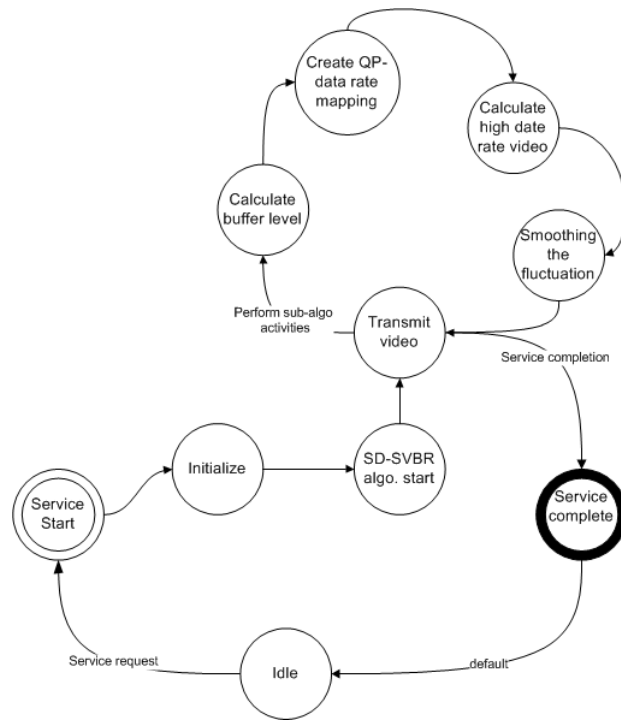
- Determining the QP value to encode the raw video data (for a GoP unit of the video data).
- Transmitting the data, which involves several sub-processes:
  - Selecting a corresponding frame size for the QP value.
  - Segmenting the frame size into TFRC packets and adding the appropriate TFRC header for transmission.
  - Transmitting the packets, then records important information about the transmission. For instance, the transmission time, arrival time, and etc.
- Calculating the bucket size.
- Determining the appropriate video GoP size for the next GoP transmission.

For the post-process part, several statistics can be calculated from the simulated traffic. These include loss rate statistics, delay statistics, user-perceived PSNR metric statistic, and redecoded transmitted video (with possible artifact effect).

### **5.3.2 Structure of Implementation: Finite State Machine**

This subsection presents the structure of SD-SVBR implementation through showing the Finite State Machine process of the mechanism, which comprises five different states with the transitions between them. The video rate control module in the server implements the SD-SVBR mechanism. The Finite State Machine also illustrates the five main processes of the SD-SVBR algorithm. Figure 5.26 shows the process model implementing the SD-SVBR mechanism.

The SD-SVBR process model, which was created and implemented in ns-2, accepts service requests from clients and forwards them to the service start module. The service process duration of each request is dependent on to the length of the video, the algorithm process time, and the transmission duration.



**Figure 5.26:** Finite State Machine Model of SD-SVBR Mechanism

The server process starts in the idle state. The idle state is where the server process would wait for an event to happen to be processed. Specifically, the server waits for the clients to request for the service. Once the service is requested by the client, the process enters the service start state. The initialized state specifies necessary initializations that are performed only one at the beginning of the operation of the SD-SVBR mechanism. Once the initialization is completed in the initialized state, a transition is made to the SD-SVBR algorithm start state.

The SD-SVBR algorithm process consists of five different states, which are Transmit video, Calculate buffer level, Create QP-video rate mapping, Calculate high video rate video, and Smoothing the fluctuation. The SD-SVBR algorithm start state begins by activating Transmit Video Process state. Here, the necessary operations in sending the video are performed. These include breaking a video GoP into video frames, then segmenting a video frame into transmission packets, after that send the packets into TFRC transmission buffer, and finally the TFRC transmission agent will

perform the necessary actions to transmit the packets.

After transmitting all the video frames in the previously-mentioned GoP video, the process enters the next state, which is Calculate Buffer Level. This calculation is done in order to prepare information for the calculation of the next GoP video rate. The next state is the Create QP-video rate mapping, where the next GoP video data is processed and the list of QP to video rate is produced. Following this, the state of Calculate high video rate is started, where the algorithm in producing a high but without overflow next GoP video rate is executed. Then, the state Smoothing the Fluctuation performs the necessary actions in order to increase the stability of the QP values, which is going to be used in the next GoP video.

After the process in the state Smoothing the fluctuation completes, the process flow returns to the Transmit video state. Here, the next GoP of the video is transmitted. The processes are repeated until the whole requested video is transmitted. Once the whole video is transmitted, the process enters the Service complete state. Then, the server returns to the idle state, where it waits for the next video request.

### **5.3.3 Implementing the Algorithm**

In implementing the algorithm, the video file is processed to produce Video Traffic Traces. These Video Traffic Traces comprise many video tracefiles with different quality and resolution. First the raw video YUV files are converted to mpeg file by using the ffmpeg tool [193], mp4.exe, and shell scripts included in the Evalvid-RA toolset. Then, tracefiles were generated for all these files and by using these tracefiles the simulation took place. The algorithm is implemented in the simulation package, which is developed using Tool Command Language (Tcl) and C++ codes.

### 5.3.3.1 Main Algorithm Flow

The main algorithm flow is depicted in Figure 5.27. Basically it follows the steps introduced by the SVBR algorithm, however, as explained previously, SD-SVBR employs almost a totally different strategy than SVBR. The way of obtaining/calculating  $R_{open}$ , estimating  $R_k$ , determining QP, and the new aspect of the algorithm, which is smoothing the QP fluctuation, followed a radically different approach.

SD-SVBR has been programmed to use the real video rate where  $R$  has been dimensioned to a high utilization of the bucket fullness level. The QP value in SD-SVBR is based on the real QP-to-video rate matching. In addition, it has to go through the mechanics of smoothing the QP fluctuations, as discussed in the previous section.

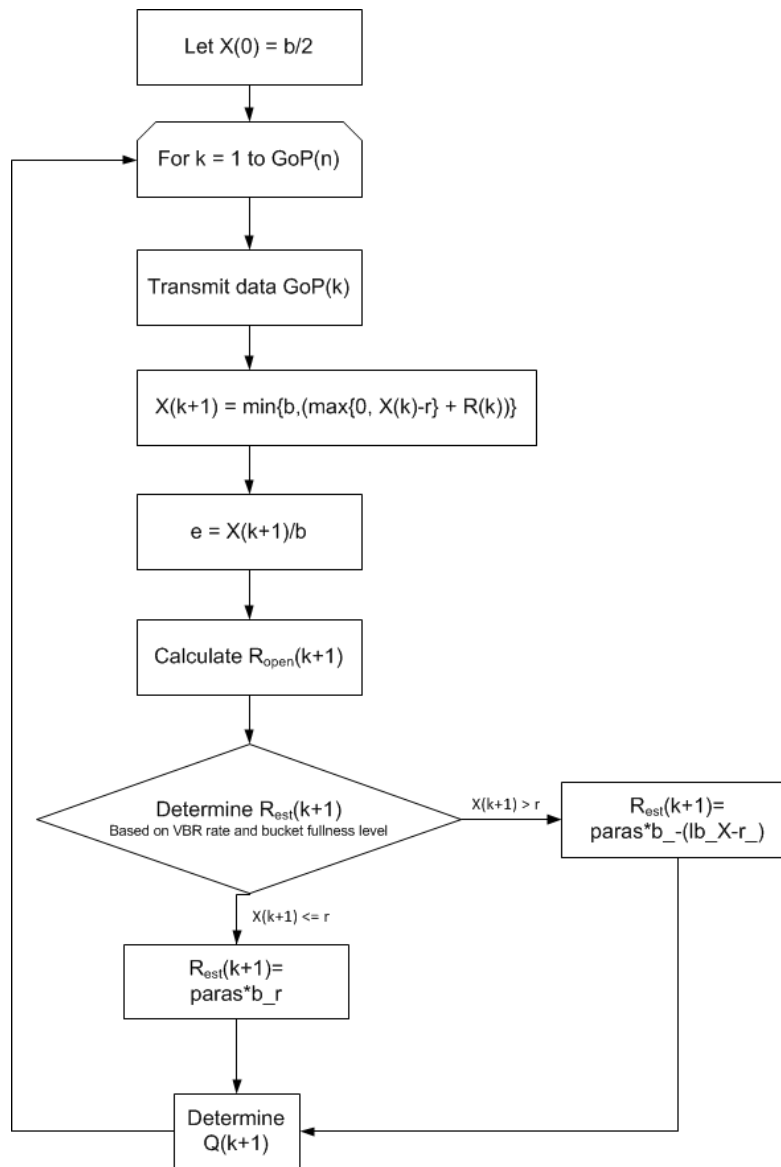
### 5.3.3.2 Tcl Program Flow

A sample of the trimmed version of the Tcl scripts is located in Appendix B. The skeleton of the flow is shown in Listing 5.1.

---

**Listing 5.1:** Skeleton of the Tcl program flow

```
1  set ns [new Simulator]
2  # Set simulation time
3      set simtime 404
4  # Set VBR video rate
5      set vbr_rate 0.33333333e6 ;# Average VBR rates for allClips
6  # Set (transmission) packet size
7      set max_fragmented_size 972 ;# MTU
8  # set frames per second (fps)
9  # Set bandwidth capacities
10 # Set Link propagation delays
```



**Figure 5.27:** The SD-SVBR main algorithm flow



```

11 # Set the size of the queue buffer
12     set queueSize1
13     [expr ($bottleneck_cap1*$queueTime)/(8*$mean_psize)]
14 # Set Global TFRC Defaults
15     Agent/TFRC set SndrType_ 1
16 # Open the files for trace generation
17 # Generate the sending node
18 # Generate the routers
19 # Generate the receiving node
20 # Define the links between the nodes
21     $ns duplex-link $send_node_1 $router_node_1
22     $access_cap $access_del DropTail
23     // we assume server will be at LAN, with 100Mbps speed rate
24 # Configure router queue monitor
25 # Set interface between TFRC Receive and the video rate controller
26     Agent/TFRC instproc tfrc_ra {bytes_per_sec backlog}
27 # Create SD-SVBR traffic sources
28     for {set i 1} {$i <= $q_variants} {incr i} {
29         set original_file_name($i) st_allclips.yuv_Q[expr $i + 1].txt
30         set original_file_id($i) [open $original_file_name($i) r]
31     }
32 # Start making GOP and Frame trace files
33     set trace_file_name video2.dat
34     set trace_file_id [open $trace_file_name w]
35     set trace_file [new vbrTraceFile_RASV]
36     $trace_file filename $trace_file_name
37 # Set TFRC (transmission/transport) agent
38     set tfrc0 [new Agent/TFRC]
39 # Set SD-SVBR application/traffic agent
40     set vbr1 [new Application/Traffic/eraVbrTrace_RASV]
41 # Link application to transport agent
42     $vbr1 attach-agent $tfrc0
43

```

```

44 #####
45 ## Simulation Main
46 #####
47 # Start the application
48     $ns at 0.010 "$vbr1 start"
49 # Capture statistics of the video transmission data
50 # Capture and calculate some utilization statistics
51     $qmon2 instvar bdrops_ bdepartures_
52 # Stop the application
53 $ns at [expr $simtime + 1.0] "finish"
54 $ns run

```

---

As for the other Tcl scripts, this skeleton consists of setting the simulation time, topology layout, topology parameters, queue size, application agent, transport agent, and running and stopping the simulation. On the other hand, since this application uses real video traffic with the video rate control algorithm, the generation of various quality of video tracefiles has to be implemented. It also has to include the video and rate control parameter setting, such as fps, bucket size, leak rate, initial QP value, and etc.

Although the video tracefiles are generated within the pre-process part, the linkages with simulation, specifically in connecting the files to the simulation program, is done in the Tcl script. With reference to Listing 5.1, the codes for executing this task is written in lines 25-29 (however a more detailed implementation is shown in Appendix B). Then the important information from the tracefiles are integrated into two main files, namely `frame_size.dat` and `gop_size.dat`.

As with the other Tcl scripts, the main purpose of these scripts is to set the simulation steps. The programming parts are mostly done in the simulation programming language, which is C++, and the built-in packages in ns2.

### 5.3.3.3 C++ Program Flow

A sample of the trimmed version of the C++ codes is located in Appendix C. The skeleton of the program flow is shown in Listing 5.2.

---

**Listing 5.2:** Skeleton of the C++ program flow

```
void vbr_rateadapt_RASV::start()
{
    init();
    timer_.resched(nextPkttime_);
}

void vbr_rateadapt_RASV::init()
{
    // read data & perform parsing
    if((f_gop = fopen("gop_size.dat", "r")) == NULL) {
        printf("can't open file %s\n", f_gop);
    }

    r_ = r_ * GoP_ / (8*fps_) ;
    // r_ now has dimensions of bytes/GOP
    b_ = b_ * GoP_ / (8*fps_); // b_ in bytes
    lb_X = b_ * 0.5; // Init buffer to half value
    lb_R = 0.0;
}

void vbr_rateadapt_RASV::timeout()
{
    //the following intructions will be executed for all the GoPs
```

```

//size_=read frame size
//calculate number of packets
x_=size_/max_; // number of complete (full sized) IP packets
y_=size_%max_; // the remaining rest part
jumPkts = x_+y_; // accumulate the packets

// Transmit the packets (if any)
if(jumPkts > 0){
    for(i=0 ; i<jumPkts; i++) {
        agent_>set_quant(Q);
        agent_>sendmsg(max_+HDR_SIZE);
    }
}

// Calculate  $X(k+1)$  @  $lb\_X(k+1)$ 
//  $lb\_X(k+1) = \min\{b, (\max\{0, lb\_X(k)-r\}+R(k))\}$ 
// Calculate bucket fullness level,  $e$ 
x_temp = lb_X/b_;
e1 = e2 = x_temp;

// Calculate  $R\_open(k)$ 
r_open = saiz_gop_w_oh[numbGops][(int)q_];

// Calculate  $R(k)$ 
//based on VBR rate categorization
//and bucket fullness level
if (lb_X > r_) {
    Rtemp = (paras*b_-(lb_X-r_));
}

//Determine  $Q(k)$ 
lb_R_wanted = Rtemp;
int jumpa=0;

```

```

        int pusingan=2;
        while (pusingan<=31 && jumpa==0) {
            saiz_gop[pusingan] =
saiz_gop_w_oh[numbGops][pusingan];
            if (saiz_gop[pusingan]<lb_R_wanted)
                jumpa=1;
            else
                pusingan++;
        }

//QP Smoothing codes
        if (pre_q==Q+1 || pre_q==Q+2) {
        if ((active==2) || (active==3))
            if (next_X <= (0.83333*b_)) {
                pre_q = (int)Q;
            }
        }

        Q = pre_q;
        numbFrames = 0;
    }

    /* figure out when to send the next one */
    nextPkttime_ = next_interval(size_);
    /* fetch both next interval and size of next frame */
    timer_.resched(nextPkttime_);
}

```

---

In actual fact, the C++ codes implements three main tasks, which are trace file parsing, SD-SVBR rate control implementation, and transporting the video traffic via TFRC transport protocol. The skeleton and C++ codes (provided in Listing

5.2 and Appendix C) only show some important parts of the SD-SVBR rate control implementation.

The trace file parsing basically reads the video trace file, parses the data, then stores the data in an array for the purpose of processing later. The data is parsed into video frame transmission time, frame size (in Bytes), frame type, and maximum size per packet. The SD-SVBR rate control implementation consists of detailed codes of SD-SVBR design and implementation. Meanwhile, the coding of video traffic transportation via the TFRC transport protocol basically implements TFRC congestion control. Since, the main part of these codes are already available in the ns2 package, the coding in the application mostly interfaces the video data with the TFRC package. Among the work that is done here are breaking up the video frames into packets (for transmission), stuffing the partial packets (because TFRC uses fixed packet sizes), and transmitting to the TFRC agent.

### **5.3.4 Allowing Dynamic GoP Size**

In the Evalvid-RA environment, the GoP size is fixed to 12 or any other fixed size which can be changed at the source code. This can be easily done by using a loop of 12 rounds (or other value) to represent 12 frames per GoP. However, if the video consists of dynamic GoP size, the application cannot process this video data because the program flows or logics will be hay-wired.

To implement video with different GoP sizes in the system, some modification needs to be done in determining the different GoP size. By using a header file or H-frame, instead of using a consistent 12 frames as an indication of the new GoP, the system can adapt for whatever frame number per GoP.

## 5.4 Summary

This chapter presented the design of a new shaped algorithm for a slight delay video application. The three main objectives of the design are to remove unwanted video rate sharp increment or decrement as of a consequence of the estimation/prediction used in the SVBR algorithm, to create a higher video rate algorithm, and to smooth the QP fluctuation. In short, the first objective is achieved by processing the next GoP video sequence and obtain the QP-to-video rate list. The second objective was attained by dimensioning the video rate to a higher utilization of the leaky-bucket, rather than based on CBR-VBR constraints as in SVBR. The last objective was accomplished by a careful measure of following the previous QP value and taking a middle value where the difference between previous QP and the intended QP value is wide.

This chapter also discussed the implementation of the SD-SVBR. This included the work environment, the Finite State Machine process model, the complete algorithm of the SD-SVBR, and the coding. The work environment is based on Evalvid-RA in ns2. The complete algorithm was implemented using Tcl and C++ codes. The skeleton of the Tcl and C++ has shown in this chapter and the samples of the scripts and codes are available in the appendices.

The next chapter will discuss the overall performance of the SD-SVBR, especially by looking at the user-perceived video performance evaluation.

# **CHAPTER SIX**

## **RESULTS AND DISCUSSIONS**

While the design and implementation have been deliberated in the previous chapter, this chapter presents a discussion on the overall result and performance of the newly created algorithm. The performance evaluation approach begins with the overall performance evaluation in terms of PSNR user-perceived video sequence, the higher video rate gained, the appropriateness of the bucket utilization, and the smoothness of the QP value. Then a deeper performance analysis is performed by zooming in at several areas in the overall chart.

### **6.1 The Overall Result**

One of the ways to evaluate the performance of the newly created algorithm is by examining the overall result. This is due to the great challenges in producing a high video rate video sequence but at the same time generating a smoother QP value. As described in the previous chapter, the challenges exists because of the conflicting result in getting a high video rate without causing fluctuation in the QP value. When the video rate is increased in a leaky bucket algorithm, the bucket fullness may be at a high level, leaving a small space for next video rate. A small space will force the algorithm to generate a low video rate. In order to produce a low video rate, the rate controller has



to use a high QP value, and this usually makes the QP value fluctuate.

On the other hand, when producing a smoother QP, the video rate may become overflowed or underflowed. Thus, by examining the overall result, the overall performance of the algorithm can be well evaluated. Another reason to evaluate the performance from an overall perspective is that in dimensioning the algorithm based on the leaky bucket approach, when the algorithm has gained a good performance in certain areas, it is difficult to obtain higher performance at other areas.

However, by a careful and robust strategy design, a better algorithm can be achieved. This design is able to produce an algorithm with a high video rate, and at the same time, a better QP stability. With a good design, the algorithm does not have to compromise between high video rate and QP fluctuation.

The average statistics of the PSNR, video rate, bucket utilization, and QP values results are listed in Table 6.1.

**Table 6.1:** The overall statistical result

Average Items	SVBR	Min SVBR	Max SVBR	SD-SVBR	Min SD	Max SD
<b>PSNR</b>	30.17	3.67	43.79	34.96	23.62	39.04
<b>video rate</b>	19228	6048	252000	20037	8064	36288
<b>Bucket utilization</b>	31682	6048	60000	41188	28208	60000
<b>QP value</b>	9.57	2	31	9.12	5	31

As described earlier, one of the ways to evaluate the performance of video quality experiments is by using the objective user-perceived evaluation, which is PSNR. On average, SD-SVBR managed to obtain almost 35dB, which is far higher than SVBR that gains only around 30dB. This difference is very big when compared with other studies. Most of the other studies just gained a fraction of PSNR increment or some other studies gain an increment on slightly higher than 1dB, as demonstrated in the following studies [194, 19]. However, this is not a very fair comparison, given that most of the mentioned studies are done in real time video application, whereas this study has an advantage of manipulating one immediate GoP information in advance.

Better still, it is believed that with a slight delay (with only one GoP delay), the increment of 5dB is considered very significant.

This impressive achievement is attributed to the triple strategy employed in the algorithm; which contributed the following results:

- i. there are almost no occurrence of overflow or underflow,
- ii. the high bucket fullness level is utilized in order to generate higher video rate for almost all of the GoPs, and
- iii. significant reduction of the QP fluctuation.

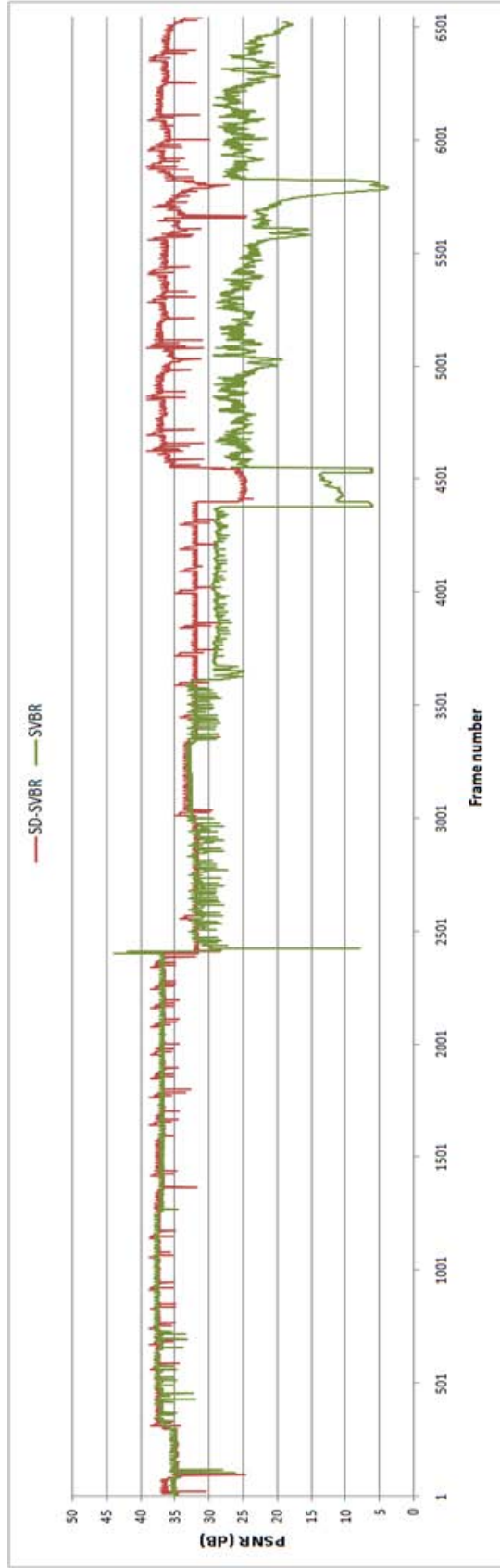
In order to analyze the SD-SVBR from the overall perspective, several charts have been generated, which will be discussed in the following subsections.

### **6.1.1 The PSNR Result**

The PSNR result is shown in Figure 6.1. Different from other charts, the PSNR chart for the y-axis is illustrated on the frame level scale. This is due to the fact that PSNR value is calculated for each frame. Since there are 6552 frames involved, it is difficult to analyze the performance in detail. The detailed performance analysis will be done by zooming into several distinct parts of the chart in the next section.

By referring to the PSNR chart in Figure 6.1, the overall PSNR gained by SD-SVBR algorithm is obviously higher than SVBR algorithm. SD-SVBR manages to gain a slightly higher PSNR from frame 1 to 2389. In other areas, referring to frames 2450-2978 and 3329-3558, the noteworthy higher PSNR is gained by SD-SVBR more than SVBR algorithm. It is worth to mention that the SVBR algorithm gains a glaring fluctuated PSNR in those areas. Those fluctuations, which are towards low PSNR value, have contributed to a lower PSNR average.

After frame 3558, the PSNR gained by the SVBR algorithm started to clearly degrade. The gap gets bigger toward the end. In the beginning, after frame 3558,



**Figure 6.1:** The overall PSNR result

the difference in PSNR value between SVBR and SD-SVBR is around 5dB. Toward the end, the gap got to be more than 10dB. These are probably due to frames lost as a result of bucket overflow and congestion in the network.

### **6.1.2 The Video Rate Result**

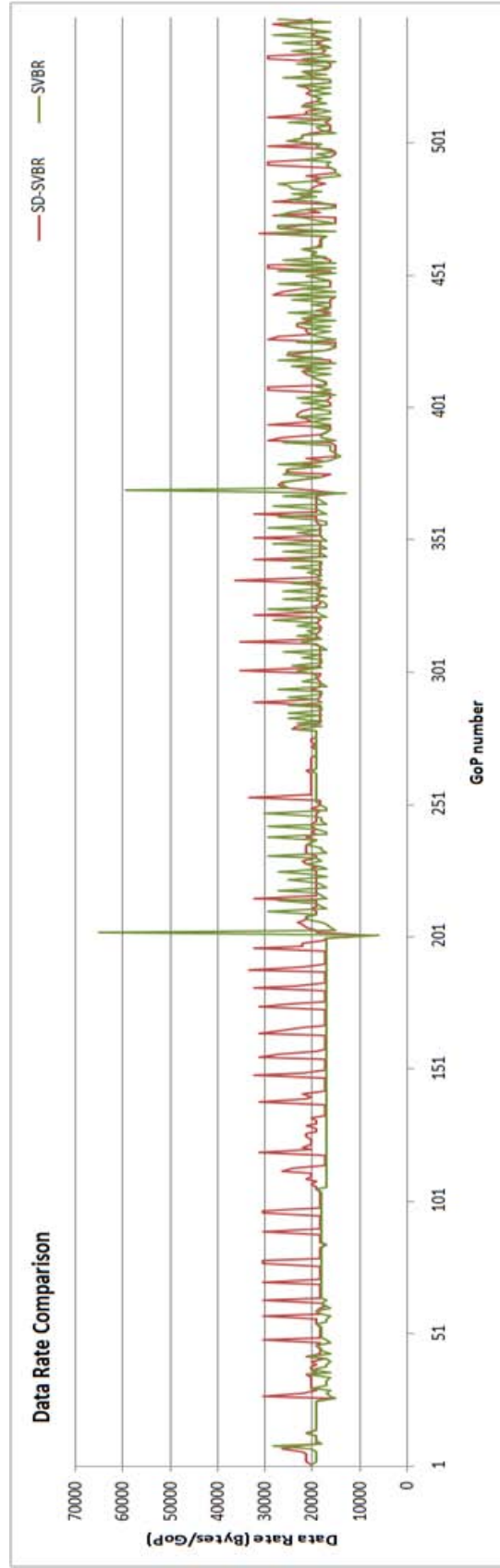
The video rate result is shown in Figure 6.2. Specifically relating to the SVBR data, GoP 201 is actually 252000 Bytes/GoP, but it is written as 65000 Bytes/GoP for the sake of chart readability and clarity. The video rate seems to fluctuate in the chart, but it is not as bad as it is displayed because the Y-axis is having more than 500 units. When the chart is zoomed in at the next section, a smoother chart can be observed.

By referring to the overall video rate chart in Figure 6.2, it cannot be simply concluded that the SD-SVBR gains a higher video rate than SVBR algorithm, except for the beginning of the chart, specifically referring to GoPs 25-200, where the SD-SVBR noticeably gains a higher video rate than SVBR.

Although the rate does not exceed the bucket size, in the situation where the video rate (input data) is higher than the leak rate which typically occurs in a high video rate video sequence, the bucket easily becomes full or overflows.

Visible fluctuation in the video rate of SD-SVBR compared to SVBR in GoPs 25-200 does not mean it has a negative performance. As discussed in Subsection 4.3.4, the fluctuation in the video rate is normal. The real negative fluctuation should be observed in the QP result. While a deeper analysis will be done in the next section, a quick observation found that the SD-SVBR experiences some level of fluctuation. However, the real consequences of this will be analyzed later.

In several other areas, specifically at GoPs 207-251 and 200-367, SVBR gains a higher video rate than SD-SVBR. As explained previously, it does not mean SVBR gains higher performance than SD-SVBR. A quick glance of the bucket utilization found that the bucket overflow occurs at those areas.



**Figure 6.2:** The overall video rate result

Finally, specifically at GoP 393 toward the end of the video sequence, the video rate fluctuation is highly visible in both algorithms. Again, this has to be crossed checked with the QP chart. This will be examined in the next section.

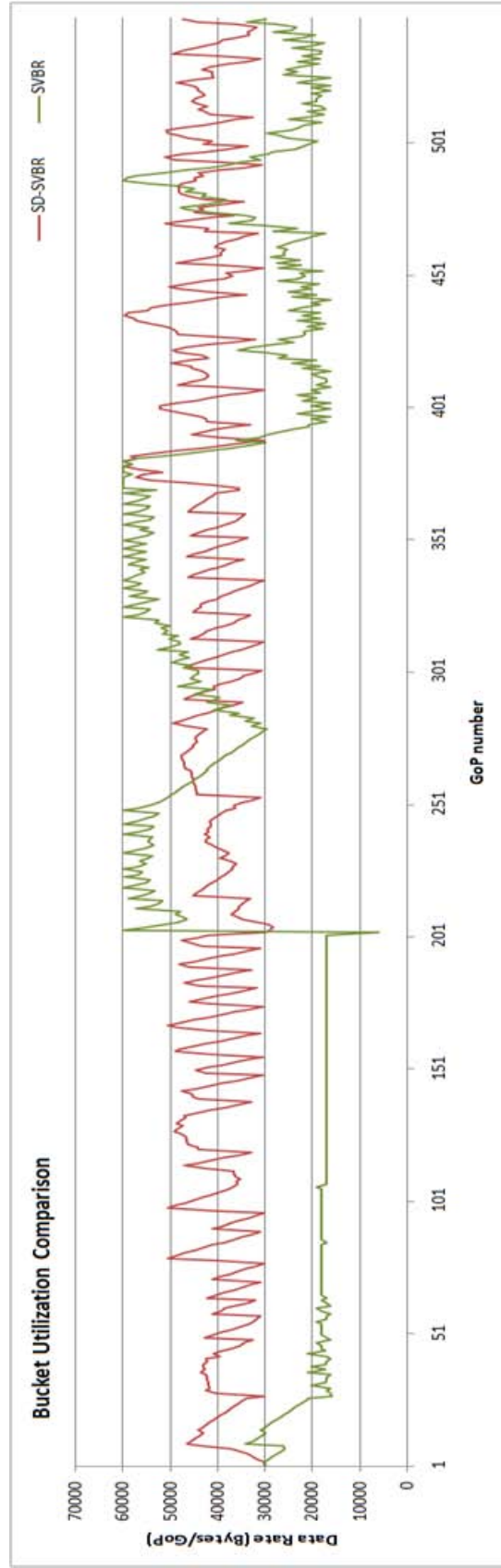
### **6.1.3 Bucket Utilization Result**

The bucket utilization result is shown in Figure 6.3. In terms of bucket utilization, the SD-SVBR algorithm utilizes quite a consistent space that is between 30000 to 50000 level. Only at around GoPs 379 and 435, SD-SVBR utilization approaches bucket size level, which is 60000 Bytes. In SVBR algorithm, the bucket utilization is not consistent, where in certain areas, the utilization is very low and at certain other areas, too high. The utilization after GoP 201 to GoP 381 experiences overflow at many GoPs. Although it is not seen in the chart, in actual transmission the overflow has occurred, which will be explained in the next subsection.

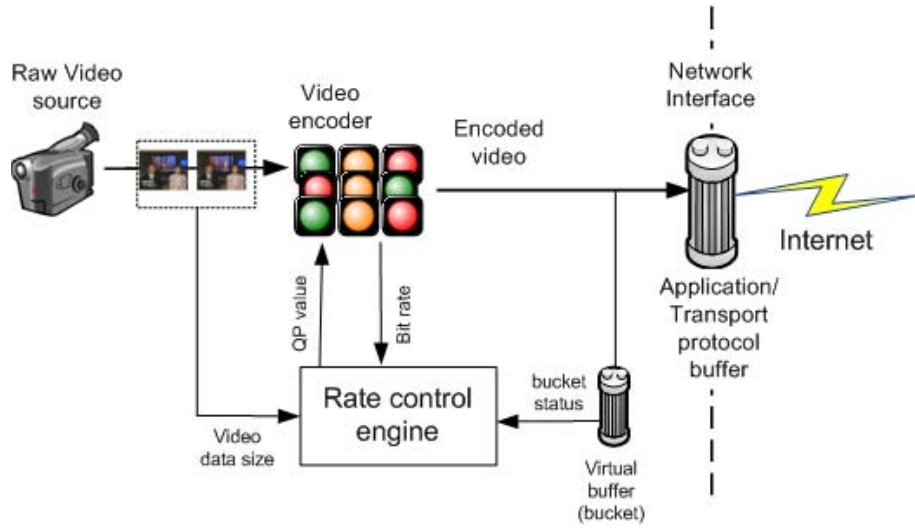
#### **6.1.3.1 Bucket Utilization**

The overall bucket utilization, as shown in Figure 6.3, does not really present the real volume of data transmitted into the network interface buffer. The chart in Figure 6.3 is based on a virtual buffer as in Figure 6.4. The virtual buffer is calculated based on the Hamdi et al. in [23] equation, which is Equation 4.2. When referring back to the Hamdi et al. algorithm, as depicted in Figure 5.1a, the equation is calculated after sending the encoded video data into the network interface. Thus, the real data in the network interface buffer might not be similar to the “mirroring” data in the virtual buffer.

The real encoded data transmitted into the network interface buffer is based on the estimated QP value, which generates overflow traffic, as explained in the previous chapter. As a matter of fact, the data or information of the data used in the rate control engine is subjected to following condition,



**Figure 6.3:** The overall bucket utilization result



**Figure 6.4:** The real bucket/buffer utilization

$$0 \leq X \leq b$$

In addition, the mathematical expression  $\max\{0, X(k) - r\}$  from Equation 4.2 produces a positive result always. While this is acceptable in order to fulfill the SVBR requirement in order to ensure that the bucket level is always between 0 and  $b$ , it is not really true in representing what is in the “real bucket” (the network interface buffer). The incorrectness might have occurred because of the following reasons:

- $X'(k) \neq X(k)$ , with the assumption that  $X'(k)$  is the data in the network interface buffer and after an overflow event occurred. After an overflow event, the imaginary data in the virtual bucket,  $X(k)$ , is equal to  $b$  or 60000. However, the  $X'(k)$  is not subjected to Equation 4.2. Thus, in the case of data overflow,  $X'(k) > b$ .
- The next GoP data will be calculated based on “imaginary” data in the bucket, instead of the real data in the network interface buffer.



Therefore, besides using the following equation,

$$X(k+1) = \min\{b, (\max\{0, X(k) - r\} + R(k))\}$$

a better representation for the network interface buffer is as follows,

$$X'(k+1) = \max\left\{0, \left(X'(k) - r + R(k)\right)\right\} \quad (6.1)$$

In Equation 6.1, the *max* function is used to ensure that the minimum data in the buffer is 0. This is needed since there will be no negative data transmission; when there is no data in the buffer, there will be no transmission. However, the maximum size of the buffer exists, which depends on the application or the transport layer buffer, and usually the size is big. However, here it is arbitrarily considered as unlimited.

By employing the mathematical expression,  $\min\{b, (\max\{0, X(k) - r\} + R(k))\}$ , of the SVBR algorithm, the expression  $\max\{0, X(k) - r\}$  will be calculated first, where the input data,  $R(k)$ , and the leak rate,  $r$ , are processed in the same GoP period. Therefore, Equation 6.1 is considered more appropriate in representing the real “Internet bucket”.

To illustrate the above mentioned calculation, Table 6.2 shows the recalculation example for GoP 485-489. For instance, after processing the video sequence for GoP 485, the data sent to the network interface buffer is 61504 (27216+54288-20000). However, in the virtual bucket it is just written 60000 (it is listed in the  $X(486)$ ), because it employs Equation 4.2.

**Table 6.2:** Recalculation example for GoP 485-489

GoP- $k$	$R(k)$	$X(k)$	$r$	$X'(k)$
485	27216	54288	20000	61504
486	19152	60000	20000	60656
487	18144	59152	20000	58800
488	14112	57296	20000	52912
489	15120	51408	20000	48032

**Table 6.3:** Recalculation example for GoP 27-34

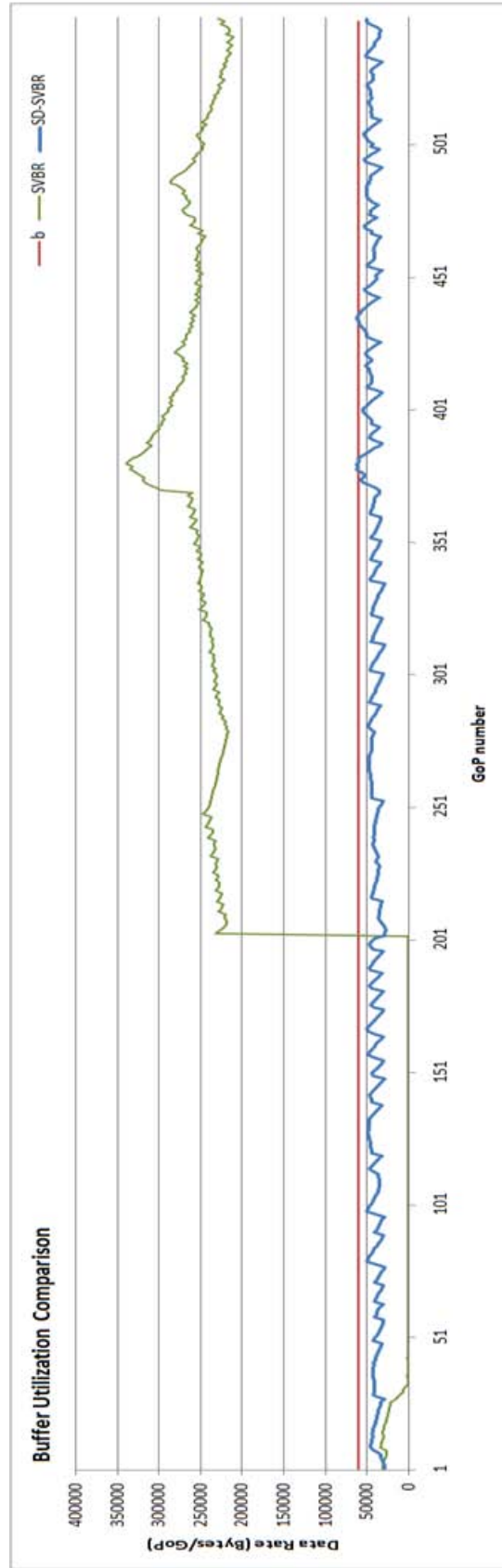
GoP- $k$	$R(k)$	$X(k)$	$r$	$X'(k)$
27	16128	16016	20000	12144
28	17136	16128	20000	9280
29	16128	17136	20000	5408
30	20160	16128	20000	5568
31	17136	20160	20000	2704
32	17136	17296	20000	0
33	17136	17136	20000	0
34	16128	17136	20000	0

The reverse implication of the “real” calculated bucket utilization is shown in Table 6.3. In the table, it illustrates that with the leak rate 20000 and the input video rate,  $R(k)$ , being less than 20000, the data fullness level in the bucket will be in a decreasing trend, as shown in column  $X'(k)$ . It might be reduced down to the minimum level, which is 0, where it happens at GoP 32-34. This demonstrates that the algorithm is at the lowest bucket utilization.

Therefore, it might be more appropriate to show the real bucket utilization chart in order to better evaluate the performance of the SD-SVBR as compared to SVBR algorithm. Figure 6.5 charts the real bucket utilization. From this figure, after transmitting an extremely high fluctuating video sequence at GoP 201, which is at 252000 Bytes/GoP, the buffer is overflowed for more than 200000 Bytes (for the sense of measurement purpose, the bucket size is only 60000 Bytes). After that GoP, the high buffer overflow is not reduced to an acceptable level until the end of the transmission.

### 6.1.3.2 Bucket Utilization in the Overall Perspective

As explained previously, the bucket utilization (refer to Figure 6.3) for SD-SVBR algorithm is high and consistent within the designated level. The lowest utilization is 30000, which is half of the bucket size. The average upper level bucket utilization is around 45000 to 50000. Although in certain areas the upper level bucket utilization is below that level, it has been designed in such a way in order to gain a smoother QP, as



**Figure 6.5:** The “real” overall bucket utilization

discussed in the previous chapter.

From the overall perspective, the bucket utilization seemed to have been increased more, for instance, to the 55000 level. Thus, more increment in the video rate can be obtained. However, this implementation has risked bucket overflow and higher QP fluctuation. Yet, with this designated bucket utilization level, the bucket overflow is still occurring, although it is at a very minimum. In terms of creating a better constant QP value, still, at some very limited regions, it is less constant than SVBR.

In the case of the bucket utilization for the SVBR algorithm, the utilization is either too low or too high. The low utilization can be seen at GoP 27 to 200. Whereas, the overly high utilization can be observed at GoPs 203 to 250, 321 to 372, and 486. The result is worse when examining the real bucket utilization, as charted in Figure 6.5.

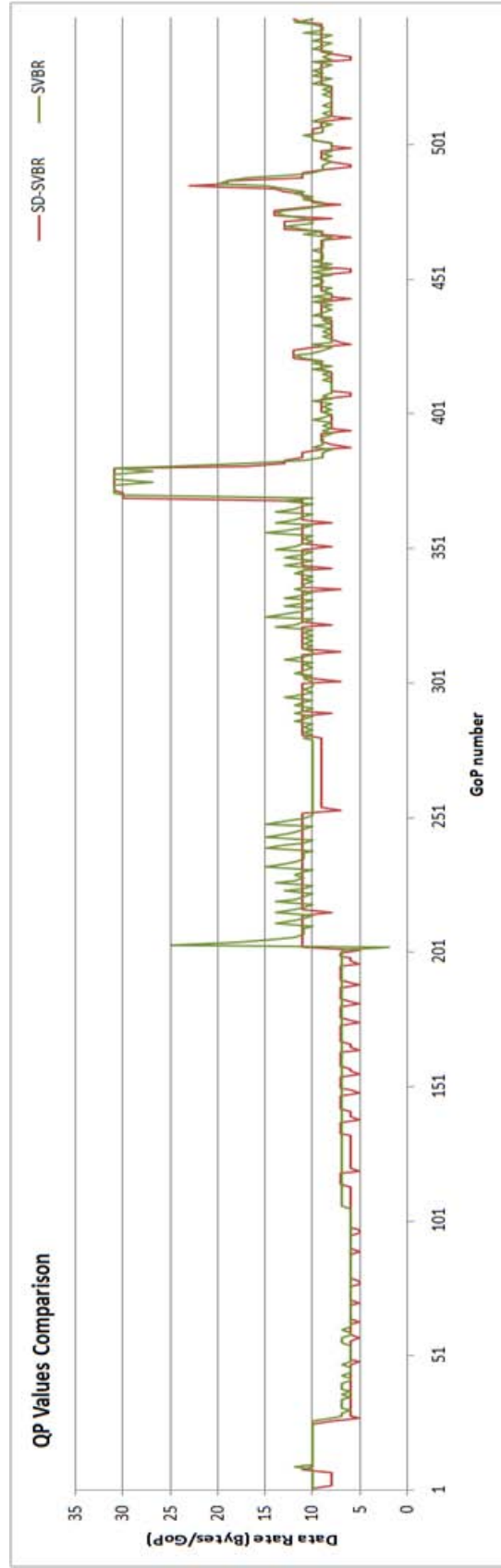
#### **6.1.4 The QP Value Result**

The QP value result is shown in Figure 6.6.

In general, SD-SVBR gets less constant QP value at GoP 27 to 200. However it gains much better constant QP on the rest of video sequences. Another comment worth mentioning is that most of the fluctuation in the SD-SVBR QP value is toward a low QP value. This means that it gains a higher video rate. However, the real implication of this will be inspected further in the next section.

### **6.2 Examining of the PSNR Result in Detail**

For the in-depth performance examination, the PSNR chart is divided into several parts. These parts are shown in Figure 6.7, 6.10, 6.15, 6.19, and 6.25. The grouping is based on the significant difference in PSNR result from the other parts. These are whether the PSNR values in one group is difference from other groups, the PSNR values in the group are almost same between SVBR and SD-SVBR algorithms, the PSNR values in



**Figure 6.6:** The overall QP value result

the group are widely difference between SVBR and SD-SVBR algorithm, the PSNR values in one algorithm are much more stable compared to the other algorithm, etc.

The frames which are involved in the grouping are as follows;

- Part 1: frame 1 to 276.
- Part 2: frame 325-2360.
- Part 3: frame 2439-3569.
- Part 4: frame 3612-4551.
- Part 5: frame 4554-6499.

In order to cross-examine the PSNR result with its equivalent video rate, bucket utilization, and QP value result, the frame to GoP mapping has to be established. This is due to the PSNR result being in frame granularity, whereas the other results are in the GoP granularity. In general, 12 frames are equal to one GoP, e.g. GoP 2 is from frame 13 to 24.

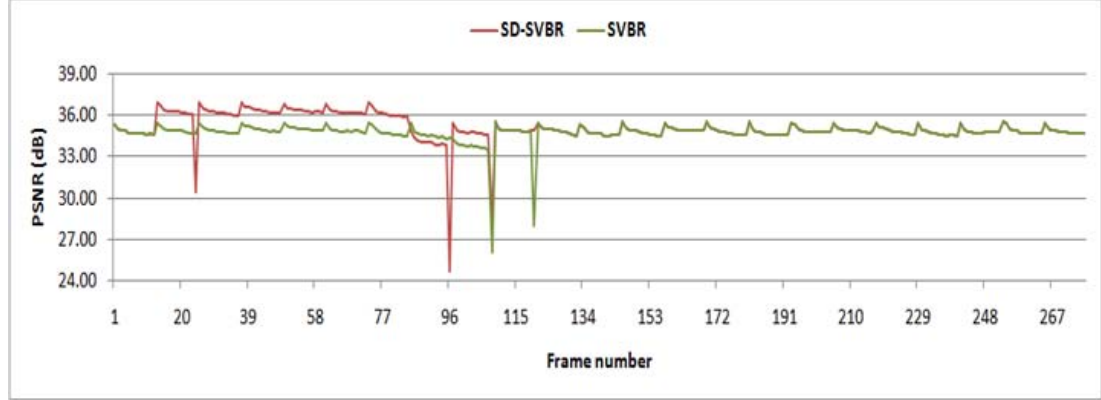
However, since the implementation of this study is also applicable for dynamic GoP size, a mapping table has to be created. Thus, for the frame to GoP mapping, it will be referred to in that table. The frame to GoP mapping table is a long table, Table 6.4 lists a sample of the mapping. The Sub Frame 0 is actually the header frame of a GoP, and it is not counted in evaluating video performance. Therefore, its Accumulate Frame number is not relevant. The Accumulate Frame number for the header frame is actually a number of the last frame from the previous GoP. The Sub Frame 1 is an I-frame and the other sub frame numbers are the P-frames.

**Table 6.4:** A sample of the frame to GoP mapping table

<b>Accumulated Frame #</b>	<b>Sub Frame #</b>	<b>GoP #</b>
4381	0	367
4382	1	367
4383	2	367
4384	3	367
4385	4	367
4386	5	367
4387	6	367
4388	7	367
4389	8	367
4390	9	367
4391	10	367
4392	11	367
4393	12	367
4393	0	368
4394	1	368
4395	2	368
4396	3	368
4397	4	368
4398	5	368
4399	6	368
4400	7	368
4401	8	368
4401	0	369
4402	1	369
4403	2	369
4404	3	369
4405	4	369
4406	5	369
4407	6	369
4408	7	369
4409	8	369
4410	9	369
4411	10	369
4412	11	369
4413	12	369

### 6.2.1 Analysis: Part I of the PSNR Result

The Part I of the PSNR result is plotted in Figure 6.7.



**Figure 6.7:** Part I: PSNR result for frames 1 to 276

When inspecting further, this part can be divided further into two sub-parts to assist the analysis (the sub-parts are distinct sub-parts of the referred chart). The first sub-part is from frame 1 to frame 108. The second sub-part is from frame 109 to the end of the Part I frames. Almost all frames in the first sub-part, the SD-SVBR gains higher PSNR value than SVBR. For the second sub-part, in almost all frames, both are gaining similar quality of PSNR.

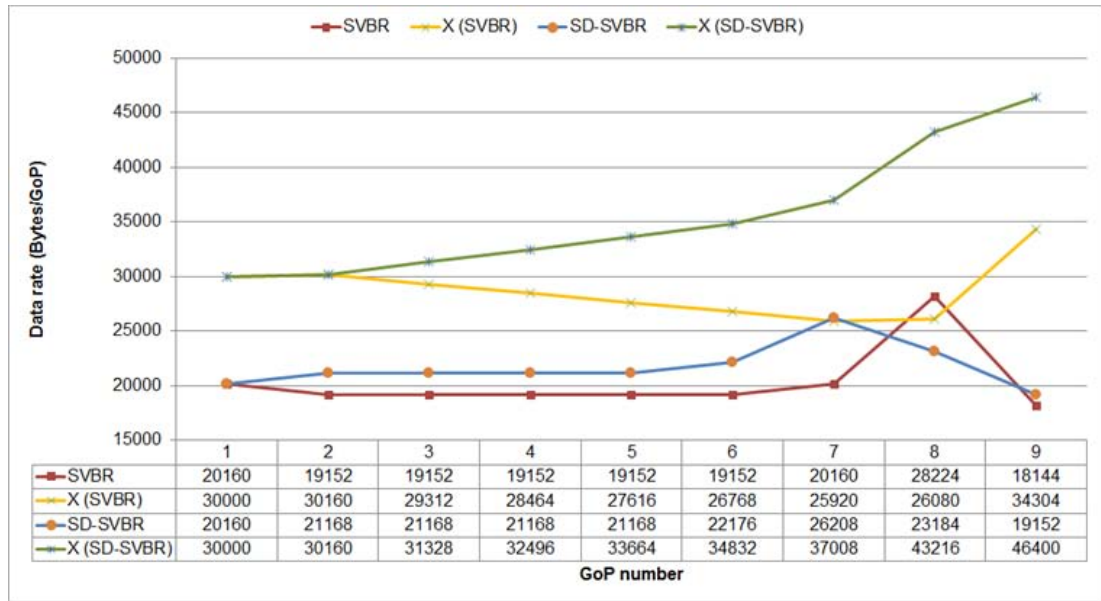
The equivalent frame to GoP is as follows,

- Frames 1 to 108 are equivalent to GoPs 1 to 9.
- Frames 109 to 276 are equivalent to GoPs 10 to 23.

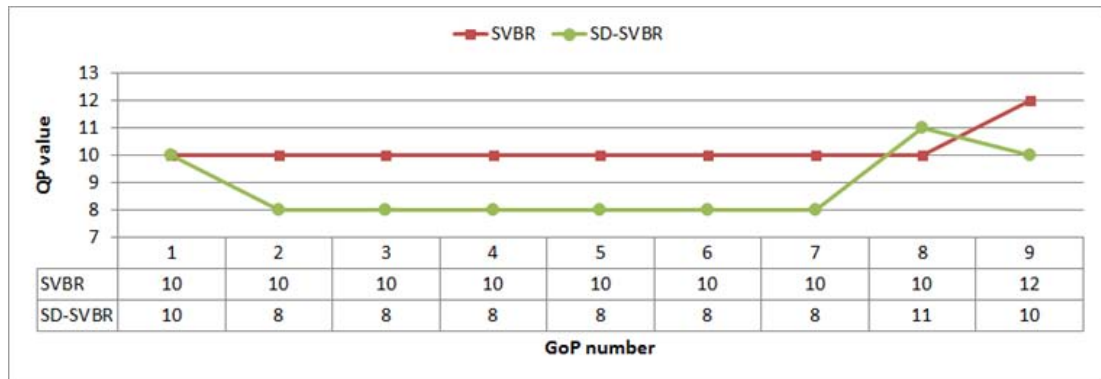
An indepth inspection was conducted to look into the equivalent of video rate, buffer utilization, and QP value. The result are shown in Figure 6.8 and 6.9.

In all metrics (video rate, bucket utilization, and QP value) for the 2nd sub-part, both algorithms give similar results (refer to Figure 6.9a and 6.9b). Except for the bucket utilization, SVBR utilizes the bucket obviously less than SD-SVBR. For this case, SVBR obtains the same performance with the SD-SVBR. All these are accurately





(a) video rate and buffer utilization

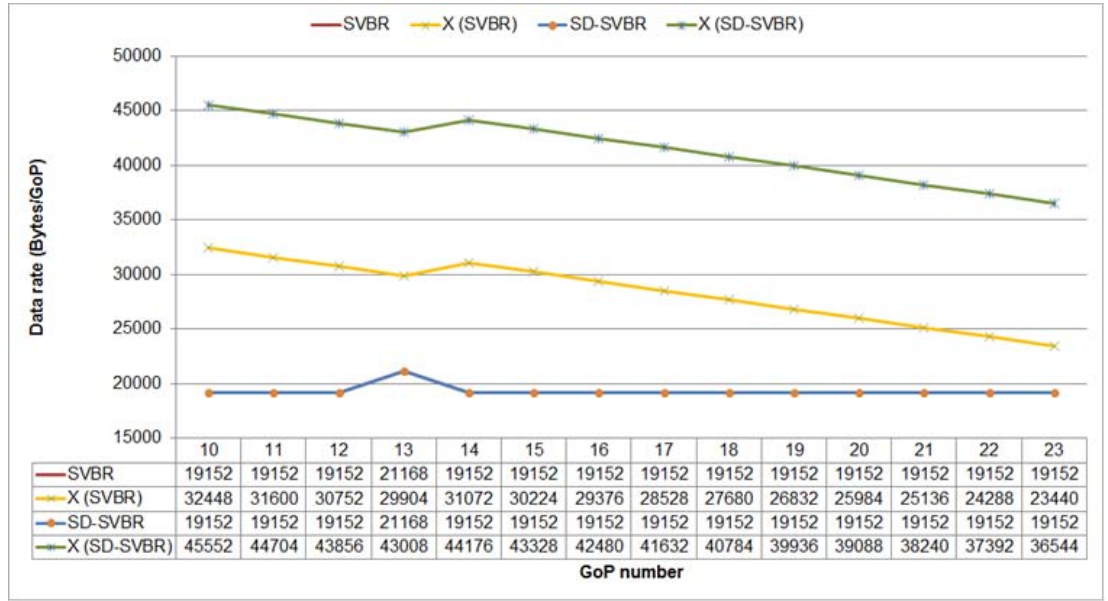


(b) QP value

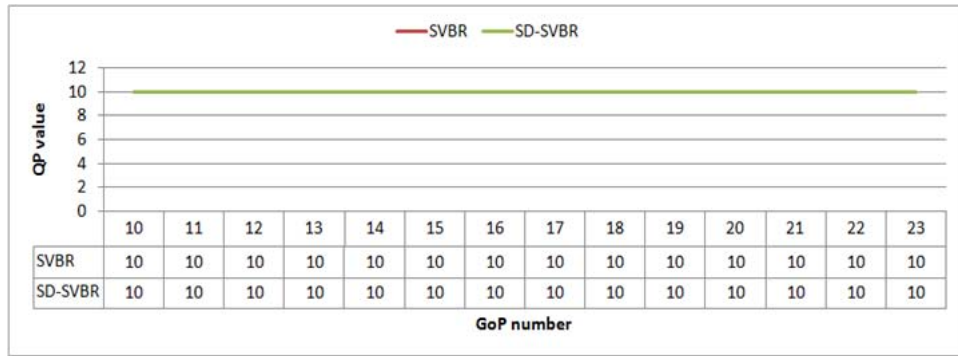
**Figure 6.8:** video rate, buffer utilization, and QP value for sub-part 1 of Part I

reflected in the equivalent frames gained for the PSNR metric. The achievement of the SVBR algorithm, even though it is equal to SD-SVBR, is something that is notable. This is due to the fact that the SVBR algorithm targets for real-time video application, where as the SD-SVBR has an advantage of a slight delay.

However, SVBR is able to obtain a similar result with SD-SVBR in the sub-part 2 because SD-SVBR has gained considerable bucket utilization, which contributes to a higher video rate than SVBR, in the sub-part 1. The higher video rate gained and better bucket utilization are reflections of what their equivalent frames got for PSNR value.



(a) video rate and buffer utilization



(b) QP value

**Figure 6.9:** video rate, buffer utilization, and QP value for sub-part 2 of Part I

As illustrated in Figure 6.8a, SD-SVBR gains a higher video rate for all GoPs, except for GoP 8. For those gains, the bucket utilization has increased up to 46400. After that, it has very limited space to increase the rate for the second sub-part of Part I, due to the SD-SVBR policy that reserves some spaces for future fluctuation or chooses a stable QP.

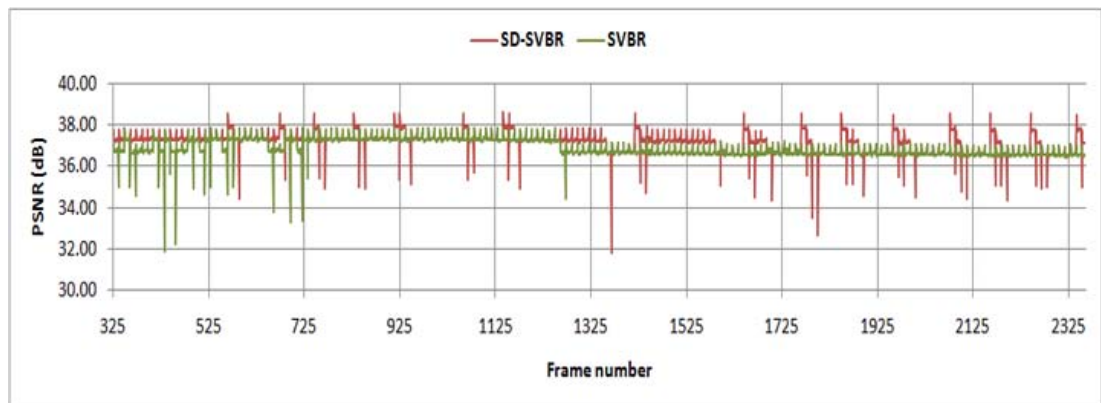
Even though SD-SVBR can be devised to increase the rate in the second sub-part of Part I, especially at the end of its GoPs, it chooses to maintain QP stability. Although the bucket utilization is decreased at the end of GoPs, which can be increased further in order to increase the video rate, it will alter the QP stability. In choosing either to

increase the video rate or to maintain the QP stability, as discussed in the previous chapter, SD-SVBR chooses QP stability if it is within a high bucket utilization level. Consequently, since it has gained a significant higher video rate in the first sub-part, it virtually cannot increase the rate higher than SVBR.

The fluctuation found in the SD-SVBR algorithm at frames 24, 96, and 120 is presented in the SVBR algorithm at frames 108 and 120. This is because both algorithms have been implemented at GoP granularity, whereas the fluctuations occurred at a certain frame in a certain GoP. For instance, frames 85-96 belongs to GoP 8, where GoP 8 obtains  $QP = 11$ , which theoretically means that all frames within this GoP have equal QP values.

## 6.2.2 Analysis: Part II of the PSNR Result

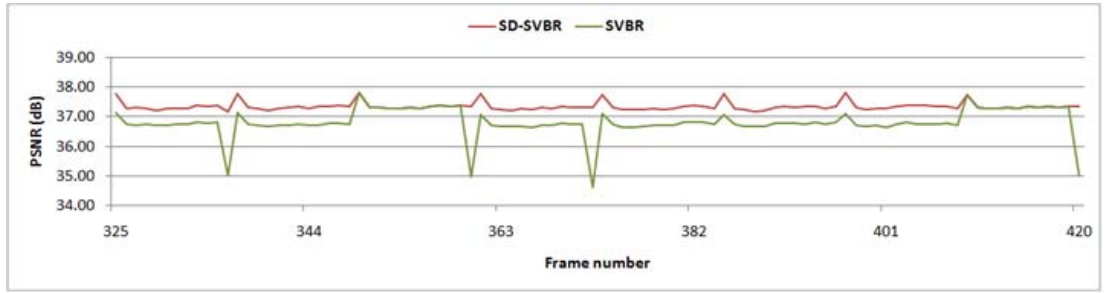
The Part II of the PSNR Result is depicted in Figure 6.10. To assist the analysis, Part II of the PSNR chart is divided further into three sub-parts, as depicted in Figure 6.11.



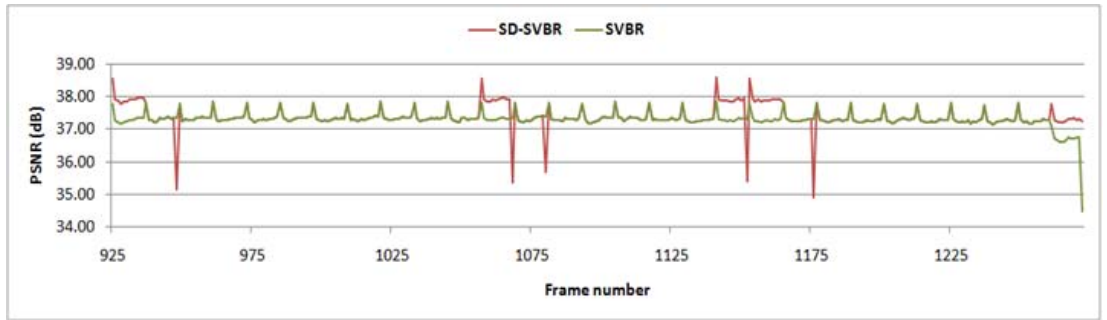
**Figure 6.10:** Part II: PSNR result for frames 325-2360

The general comments of these sub-parts are as follows,

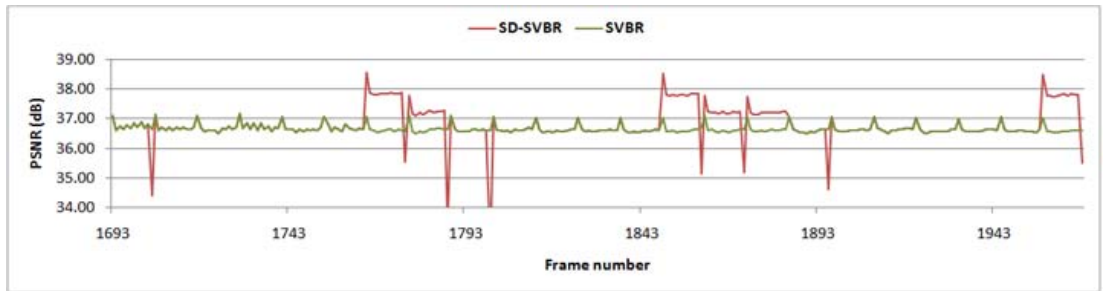
- In sub-part 1, which is for frames 325-420 as depicted in Figure 6.11a, SD-SVBR has gained a higher PSNR value than SVBR in most frames.
- In sub-part 2, which is for frames 925-1272 as shown in Figure 6.11b, SD-SVBR



(a) PSNR for frames 325-420



(b) PSNR for frames 925-1272



(c) PSNR for frames 1693-1968

**Figure 6.11:** The PSNR sub-parts for the Part II chart

gains similar quality as SVBR in many frames. However, SD-SVBR still gains higher PSNR value in several groups of frames. Actually, each group of frames is equal to one GoP. For instance, the first group consists of frames 925 to 936, which are frames for GoP 78. Similarly, the second group of frames, which are frames 1057-1068, are the frames for GoP 89. This manifests well with this study that has implementation at the GoP level. The corresponding result for the GoP level in terms of the video rate, bucket utilization, and the QP value will be elaborated later.

- In sub-part 3, which is for frames 1693-1968 as presented in Figure 6.11c, SD-SVBR gains better performance than SVBR in almost a similar fashion of sub-part 2. The higher performance is observed in more groups of frames.

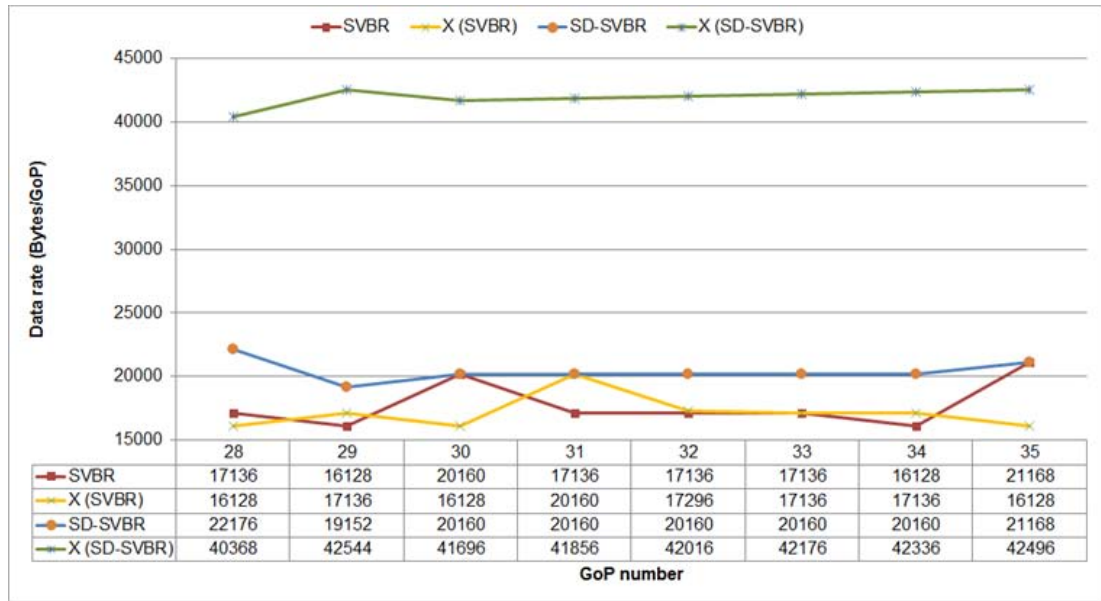
### **Analysis of the Corresponding video rate, Buffer Utilization and QP Value**

To further examine these PSNR results, the corresponding video rate, buffer utilization, and QP value are plotted. As described in the previous subsection, the corresponding frames need to be mapped with their associate GoPs. These associations are as follows:

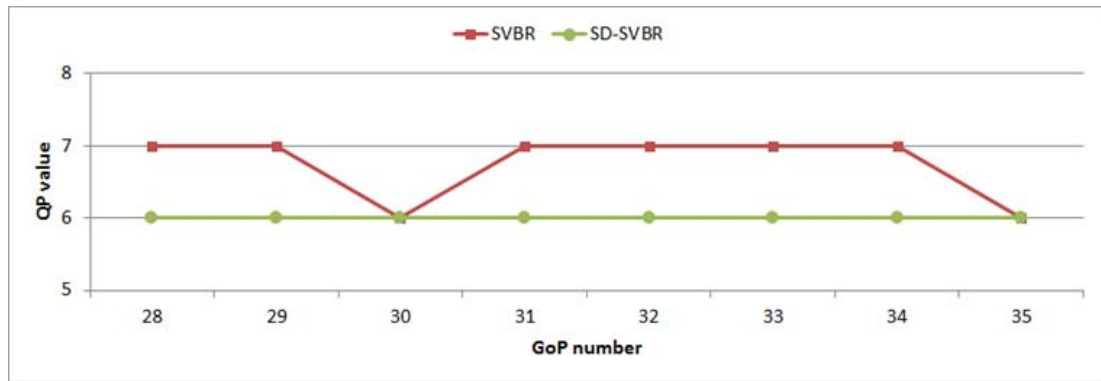
- Frames 325 to 420 are equivalent to GoPs 28 to 35. The corresponding video rate, buffer utilization, and QP value are shown in Figure 6.12.
- Frames 925 to 1272 are equivalent to GoPs 78 to 106. The corresponding video rate, buffer utilization, and QP value are plotted in Figure 6.13.
- Frames 1693 to 1968 are equivalent to GoPs 142 to 164. The corresponding video rate, buffer utilization, and QP value are depicted in Figure 6.14.

For sub-part 1, the higher PSNR gained by SD-SVBR algorithm is reflected well in terms of higher video rate and lower QP value gained, as depicted in Figure 6.12a and 6.12b respectively. In particular to the video rate and bucket utilization, as illustrated in Figure 6.12a, SD-SVBR has well utilized the bucket to the devised level. Although the video rate gained is not high enough, that is the only achievable video rate that can be gained with the limited space available in the bucket. However, it is still meaningful since the video rate is higher than the SVBR algorithm. This performance is more significant when inspecting the gained QP value. SD-SVBR is able to yield a very stable QP value, and it is lower than SVBR.

The PSNR value gained by SD-SVBR in sub-part 2 of Part II chart (refer to Figure 6.11b), as described previously, is almost similar to the value gained by SVBR.



(a) video rate for GoPs 29-35

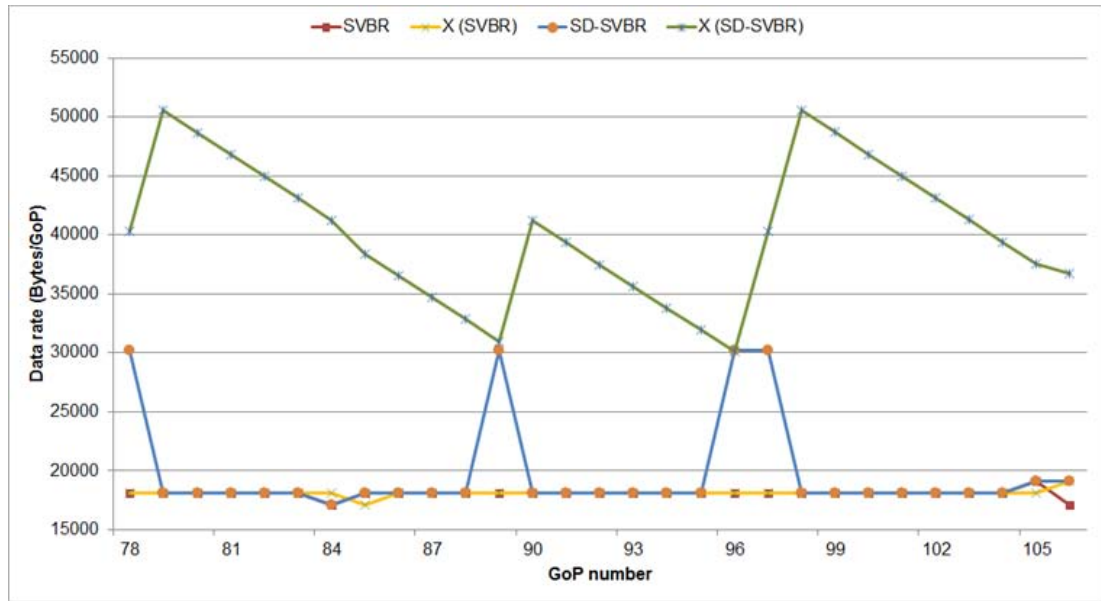


(b) QP values for GoPs 29-35

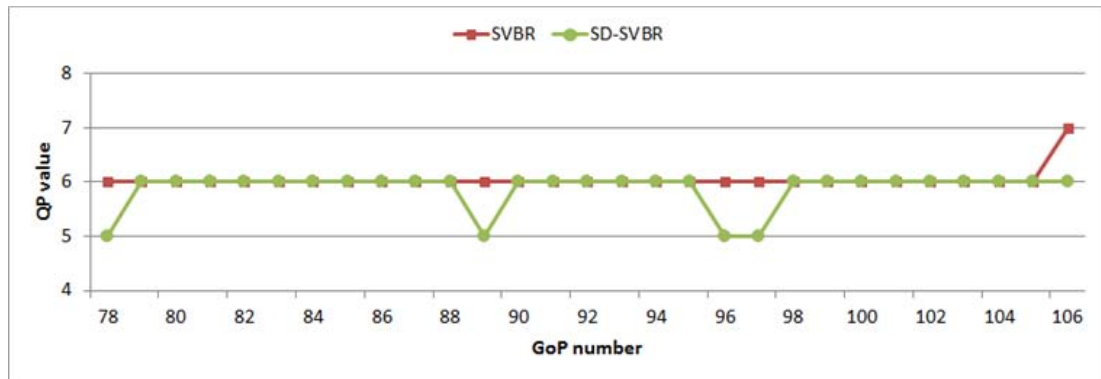
**Figure 6.12:** The video rate and QP value for sub-part 1 of Part II chart

However, in a few groups of frames, SD-SVBR gains higher performance than SVBR. In the context of the video rate and QP value, as shown in Figures 6.13a and 6.13b, it reflects what is gained in the PSNR chart.

When examining further the bucket utilization for several GoPs, as depicted in Figure 6.13a, the bucket utilization is decreased below 40000 level. The bucket utilization can be increased in order to gain a higher video rate at those GoPs. However this will create a fluctuation QP value. It seems that it might be possible to produce an average higher video rate and smoother QP fluctuation. However, that is not possible



(a) video rate for GoPs 78-106



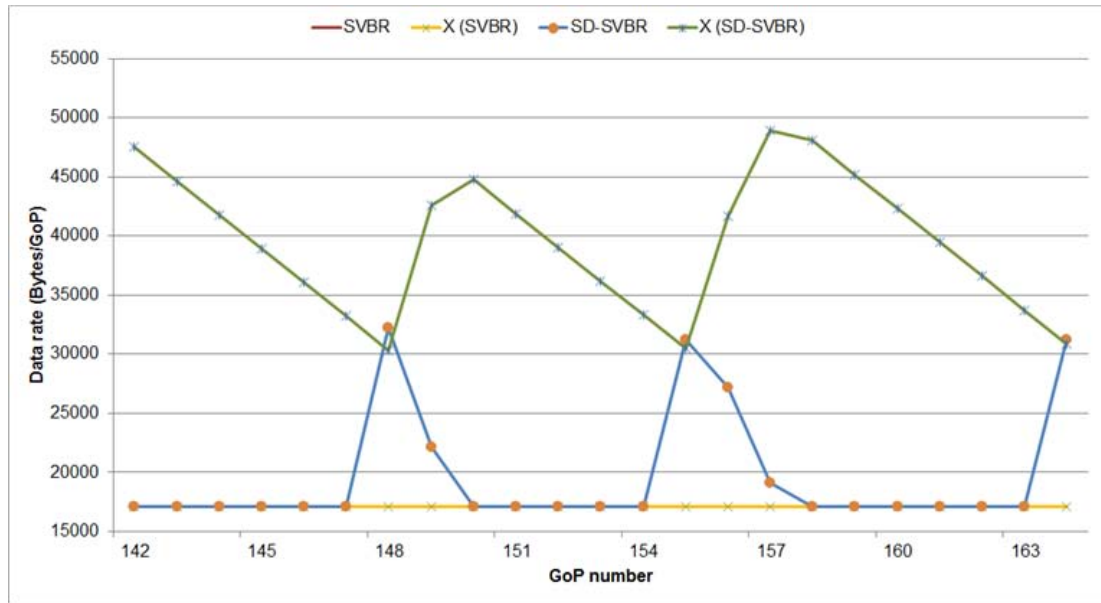
(b) QP value for GoPs 78-106

**Figure 6.13:** The video rate and QP value for sub-part 2 of Part II chart

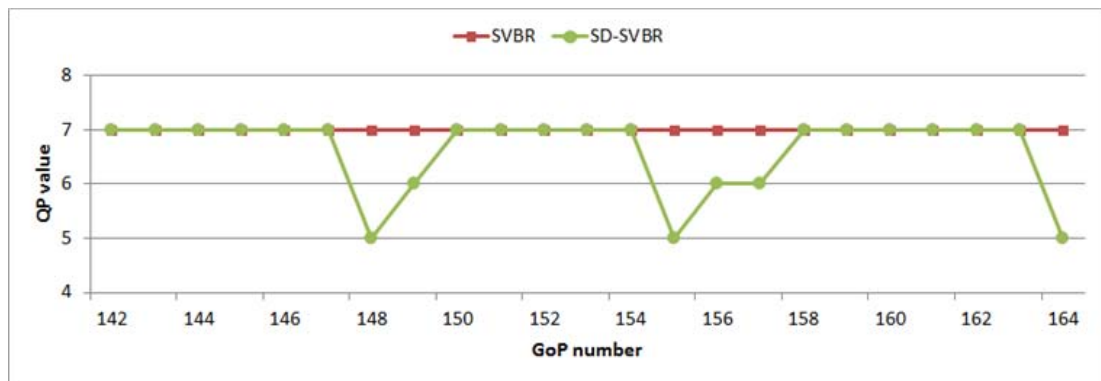
in the one-GoP slight delay video application, since several GoP information is not available beforehand.

The PSNR value gained by SD-SVBR in sub-part 3 of Part II chart (refer to Figure 6.11c) is almost similar performance with the sub-part 2 of Part II chart. SD-SVBR got a relatively higher video rate and lower QP value than what has been gained in sub-part 2. Again, this also reflects well in what is shown in the PSNR chart.





(a) video rate for GoPs 142-164



(b) QP value for GoPs 142-164

**Figure 6.14:** The video rate and QP value for sub-part 3 of Part II chart

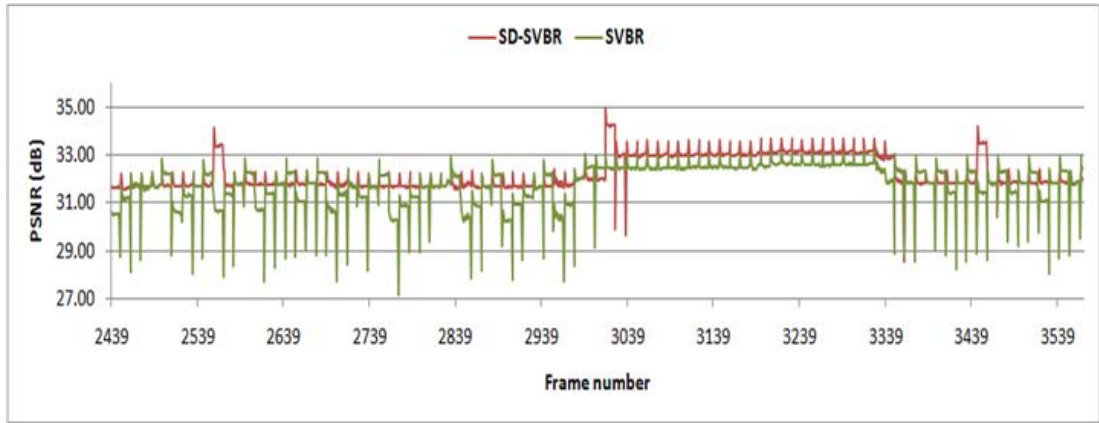
### 6.2.3 Analysis: Part III of the PSNR Result

The Part III of the PSNR result is depicted in Figure 6.15.

To assist the Part III analysis of the PSNR chart, two sub-parts are extracted from the chart. The last portion of the PSNR chart is in a similar trend as the first sub-part. Thus, only two sub-parts are going to be examined in-depth and these two sub-parts are as follows:

- The first sub-part is from frame 2594 to frame 2748. The equivalent GoPs are from 218 to 230.

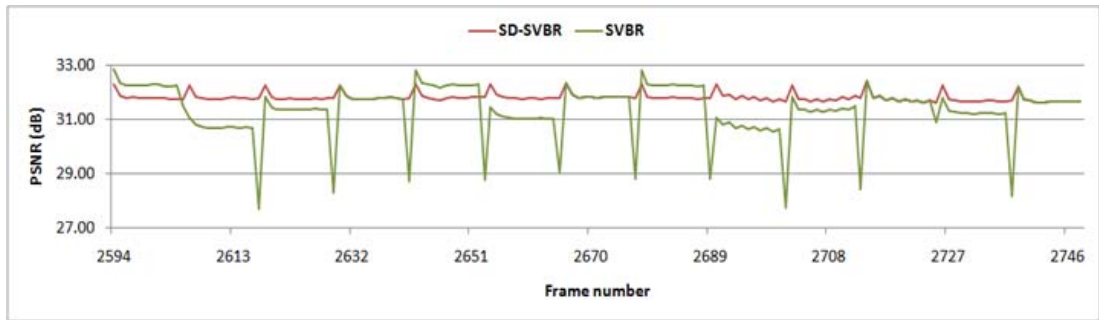




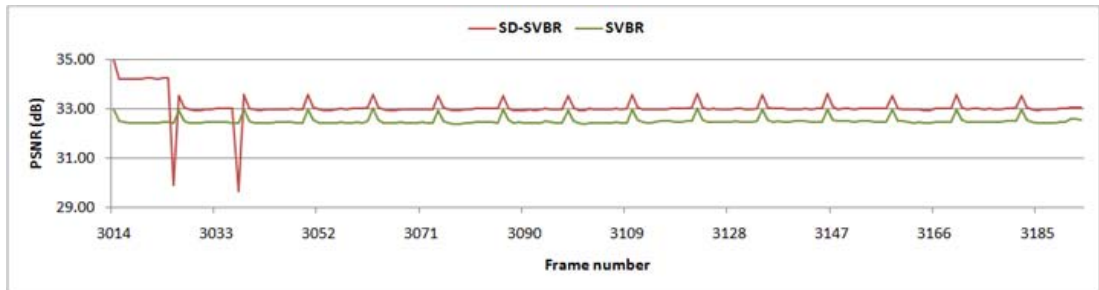
**Figure 6.15:** Part III: PSNR result for frames 2439-3569

- The second sub-part is from frame 3014 to frame 3193. The equivalent GoPs are from 253 to 267.

The two sub-parts are plotted in Figure 6.16a and 6.16b.



**(a)** PSNR for frames 2594-2748



**(b)** PSNR for frames 3014-3193

**Figure 6.16:** The PSNR sub-parts for the Part III chart

In the first sub-part, SD-SVBR gains approximately similarly to SVBR. In some groups of frames, SD-SVBR gains higher performance than SVBR, but in some other

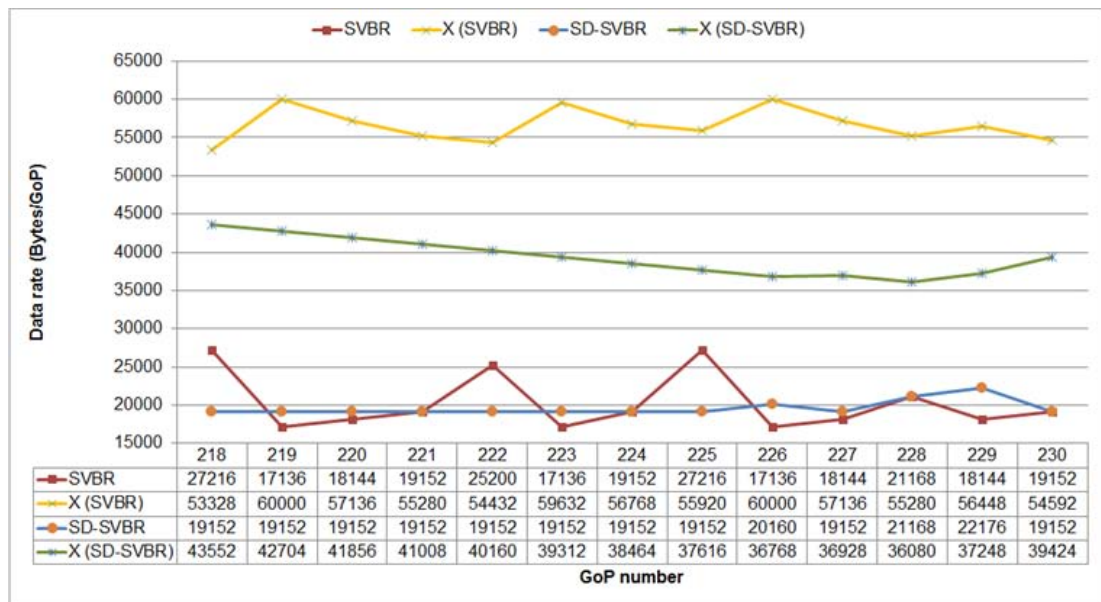
groups, SVBR obtains better performance than SD-SVBR. However, it is interesting to cross-examine this phenomenon with its corresponding video rate, bucket utilization, and QP value, which will be explained later, on the reason why SD-SVBR is not able to gain better performance here. For the second sub-part, SD-SVBR gains a higher PSNR value for almost all frames, except for the two intermittent frames.

### **Analysis of the Corresponding video rate, Buffer Utilization and QP Value**

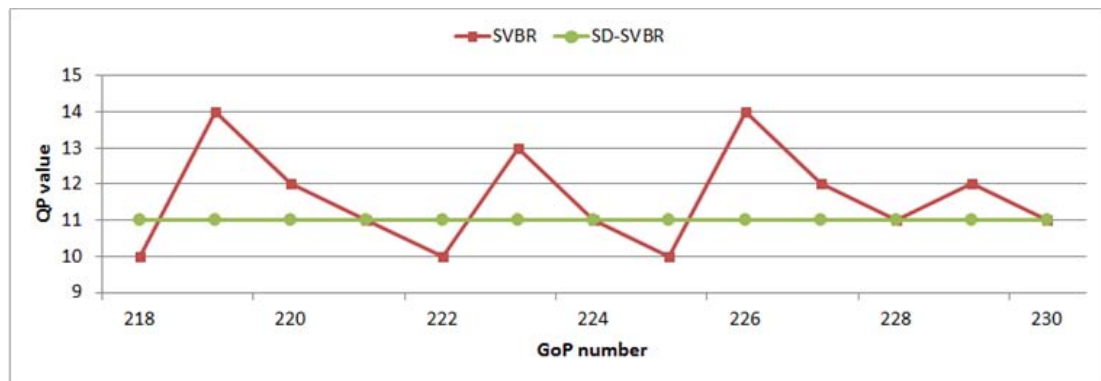
The corresponding video rate, bucket utilization, and QP value for the two sub-parts are shown in Figure 6.17 and 6.18. As mentioned previously, it is interesting to know why SD-SVBR does not gain better performance in the first sub-part. When inspecting the bucket utilization for the SD-SVBR algorithm, it seems that it has used the space in the bucket as it was designed to. The decreasing trend in bucket utilization is due to its design objective, which is to choose a smoother QP while it is still in the utilization mode. As discussed in the previous chapter, when it chooses to follow the previous QP value, it might compromise its high video rate objective. With that bucket utilization, the video rate is not at its optimum level.

In the SVBR algorithm, the gained video rate is fluctuated. Sometimes, it gains higher video rate than SD-SVBR and some other times it gains lower video rate than SD-SVBR, while at some GoPs, it gains a similar video rate as SD-SVBR. These reflect what have been gained in the corresponding PSNR chart. However, SVBR attains that video rate with a potentially fatal weakness. During several occasions, its bucket utilization is overflowed. These can be observed after the data transmission for GoPs 218 and 225. After transmitting data for GoP 218, the actual buffer fullness level is  $27216 + 53328 - 20000 = 60544$ . Also, after transmitting data for Gop 225, the actual buffer fullness is  $27216 + 55920 - 20000 = 63136$ .

How does the SVBR algorithm gains a high PSNR value with the bucket overflow? By understanding the mechanics of the rate control algorithm, one might be able to



(a) video rate for GoPs 218-230

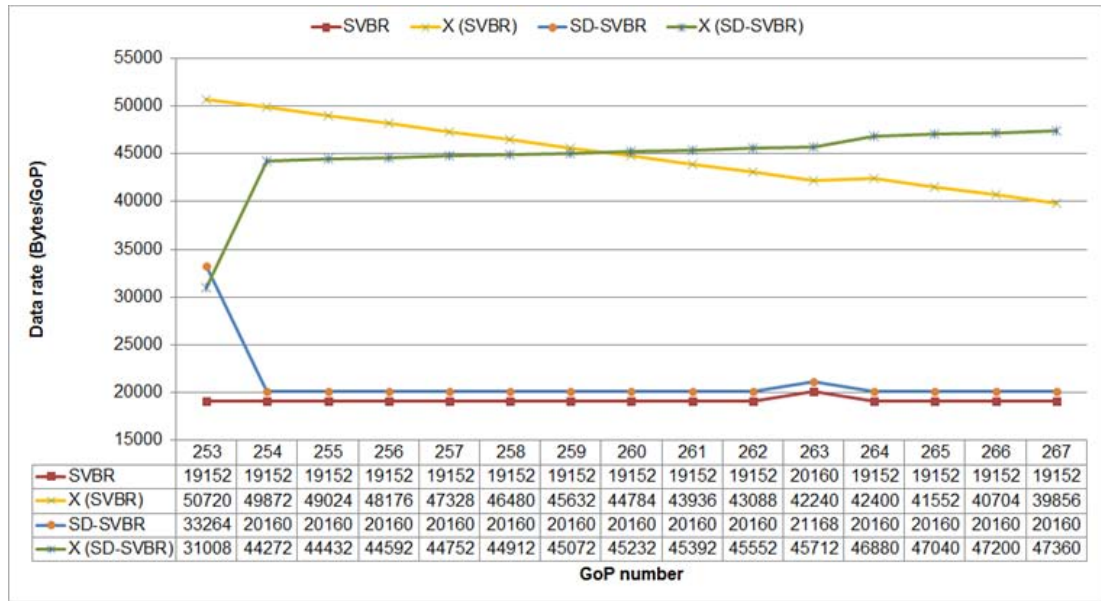


(b) QP for GoPs 218-230

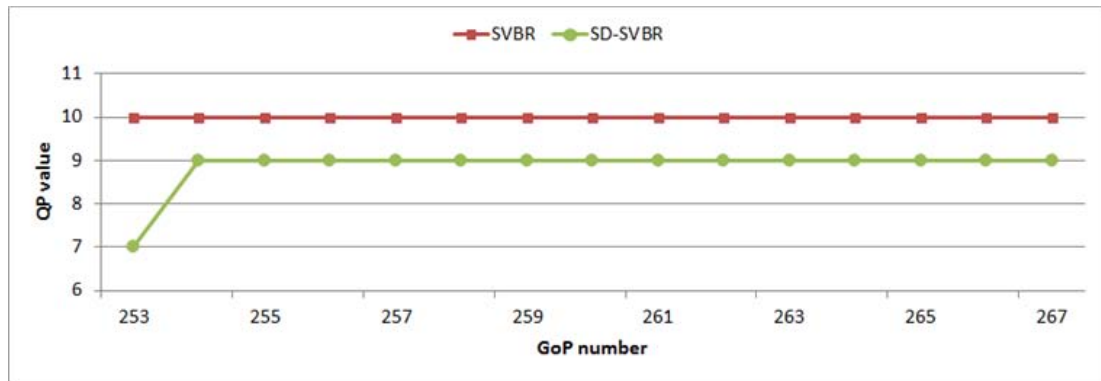
**Figure 6.17:** The video rate and QP value for sub-part 1 of Part III chart

answer this question. As elaborated in Subsection 6.1.3.1, the rate control will feed a QP value to the video encoder, then the video encoder will compress the raw video data into an encoded video. The encoded video information will be fed back into the rate control engine to calculate QP for the next GoP. However, the real encoded video sequence data will be temporarily stored into the network interface buffer for transmission.

The bucket size is designed to imitate the maximum size of a network bandwidth rather than the maximum of the network interface buffer size. Thus, although the



(a) video rate for GoPs 253-267



(b) QP for GoPs 253-267

**Figure 6.18:** The video rate and QP value for sub-part 2 of Part III chart

encoded data might be overflowed at the virtual bucket, the real encoded data is not stored there. It is directly sent to the network interface buffer, which is usually implemented with a large storage. When the data are available at the network interface buffer and ready for transmission, the transport protocol will take control of the transmission. This transport protocol works independently with what has been produced by the application layer rate control algorithm, as stated in [175].

The implication of the above-mentioned issue is that the data in the buffer will be sent to the receiver according to the transport layer rule. However, with the “bandwidth

overflow data amount” at the network interface buffer, the data are not being able to be sent promptly.

There are at least two possibilities here; firstly, it will be sent according to the transport layer protocol, but it will take a longer time for the whole data to be transmitted. The second possibility is that the application designer can opt to drop some packets/frames. By dropping some overloaded packets/frames, the other frames can be sent on time. Thus, its application really depends on the video application objective. Although, the first approach sounds more interesting – a delay without packet drops – when overload occurs in the network, most of the frames will be sent at delayed time and the possibility of packets/frames dropped is higher, which can lead to a congestion collapse network.

In the implementation of this study, the first possibility is executed. Therefore, the packets/frames, even in the overflow event, are can still be sent to the receiver. Most of the packets/frames will be received by the receiver but in at delayed time. However, some packets/frames might be lost due to traffic congestion. The result of the experiments in this study has shown that several frames have been lost. Thus, when an overflow event occurs, it does not mean that the immediate frames will be lost. It might affect several frames after that. Thus, the immediate frames of the overflow bucket might still obtain high PSNR, but it may degrade PSNR for some future frames. This scenario happens in this study.

Besides the overload problem generated by the SVBR algorithm, it also generates a less stable QP value than SD-SVBR algorithm. Thus, although the PSNR and the video rate gained by SVBR are more or less than the SD-SVBR algorithm, it is still very much problematic in terms of buffer overflow and less stable QP value.

For the second sub-part of Part III result, SD-SVBR has gained better performance than SVBR. This is reflected well with the better gains in PSNR value, higher video rate, higher bucket utilization, and lower and acceptable stable QP values.

#### 6.2.4 Analysis: Part IV of the PSNR Result

The Part IV of the PSNR result is depicted in Figure 6.19.



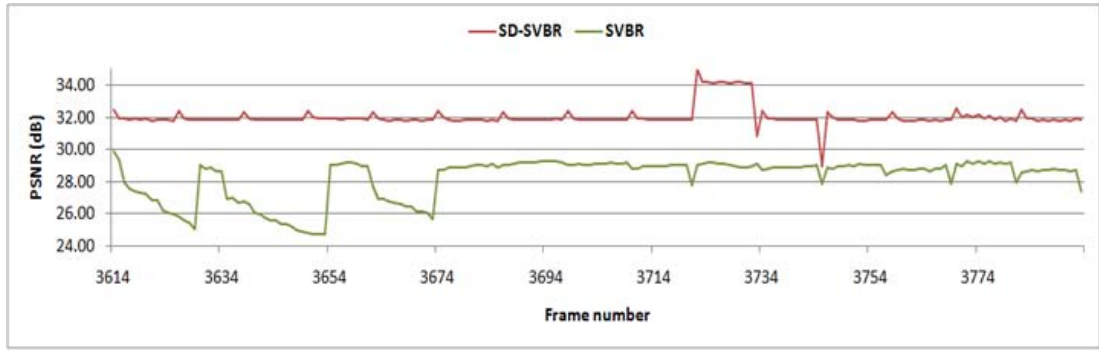
**Figure 6.19:** Part IV: PSNR result for frames 3612-4551

To assist in analyzing Part IV of the PSNR chart (refer Figure 6.19), two sub-parts are extracted from the chart. These two sub-parts are as follows:

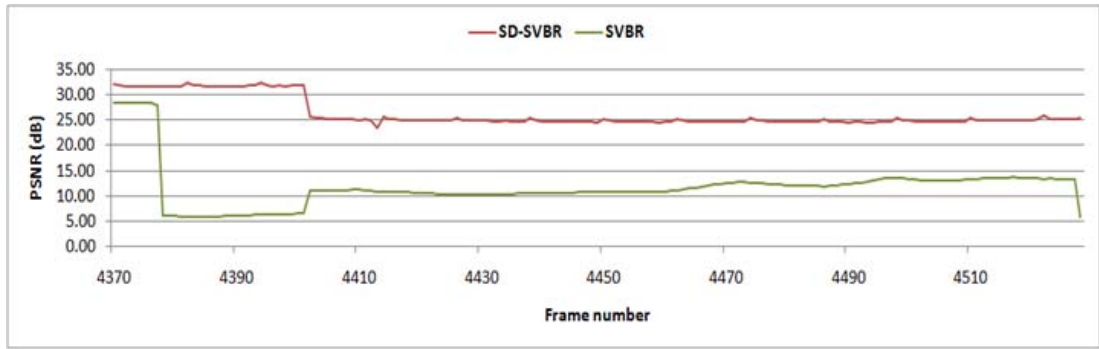
- The first sub-part is from frame 3614 to frame 3793. The equivalent GoPs are from 303 to 317.
- The second sub-part is from frame 4370 to frame 4528. The equivalent GoPs are from 366 to 379.

The two sub-parts are charted in Figure 6.20.

In the first figure, Figure 6.20a, the difference in PSNR value between SD-SVBR and SVBR algorithm is at least 3dB and in the second figure, Figure 6.20b, the difference is more than 10dB, which is very high. This high difference is not due to low video rate gained. When inspecting at the corresponding video rate, as charted in Figure 6.21a, the SVBR algorithm gains a higher video rate than SD-SVBR in many of the GoPs. Even, in terms of bucket utilization, SVBR utilizes the bucket much better than SD-SVBR. Nevertheless, the bucket utilization by the SD-SVBR is still within its design objective.



(a) PSNR for frames 3614-3793



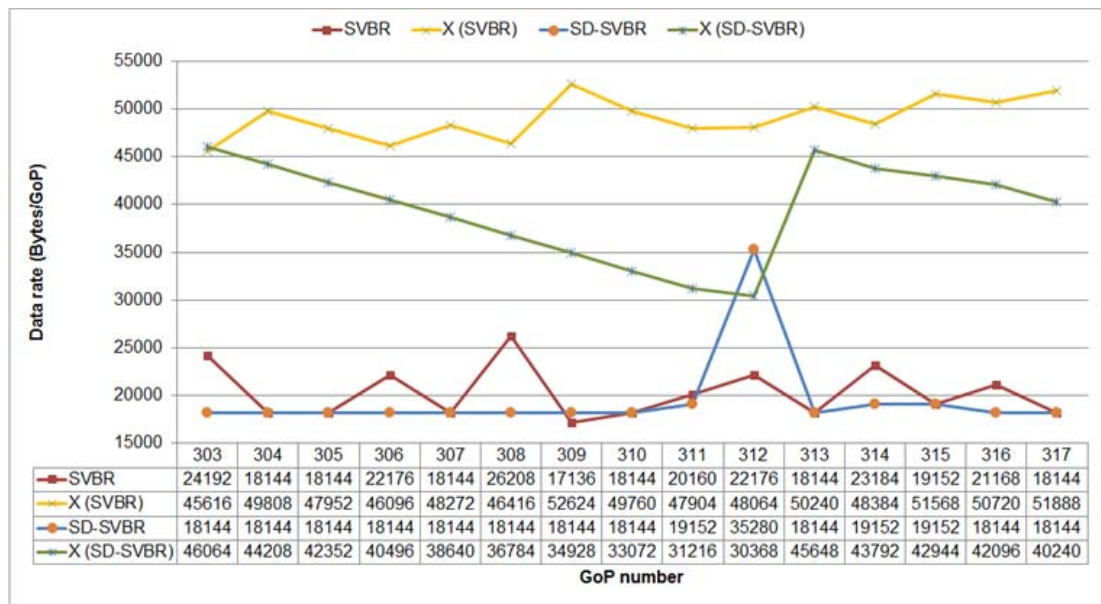
(b) PSNR for frames 4370-4528

**Figure 6.20:** The PSNR sub-parts for the Part IV chart

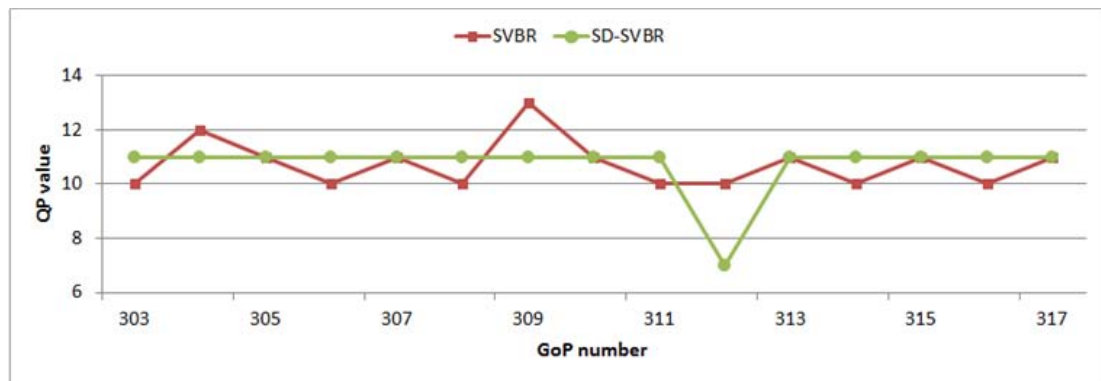
However, the high video rate and better bucket utilization gained by SVBR, has come with a high price in terms of QP instability. As shown in Figure 6.21b, the QP value for the SVBR algorithm is very much fluctuated, and it is almost unstable. Whereas, in a very much contrasting fashion, the SD-SVBR algorithm gains very good QP stability.

Actually, the QP stability in SD-SVBR algorithm is spanned across many GoPs either before or after sub-part 1 GoPs. The SD-SVBR QP stability in these groups shows a very clear contradiction when compared to what is gained by the SVBR algorithm. This excellent achievement is clearly demonstrated in Figure 6.22. In the SVBR algorithm, as charted in Figures 6.22a and 6.22b, the change in QP value occurs in every one or two GoPs, whereas in the SD-SVBR algorithm, the change is only presented in a distance of around 10 GoPs.





(a) video rate for GoPs 303-317



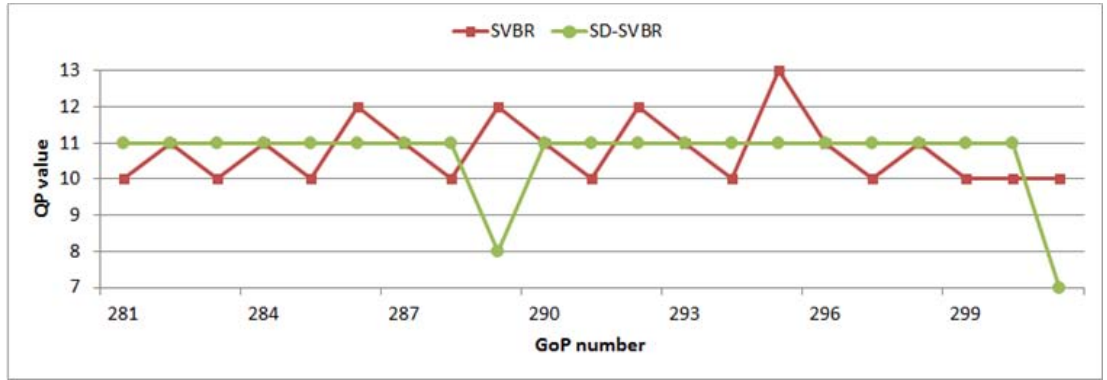
(b) QP for GoPs 303-317

**Figure 6.21:** The video rate and QP value for sub-part 1 of Part IV chart

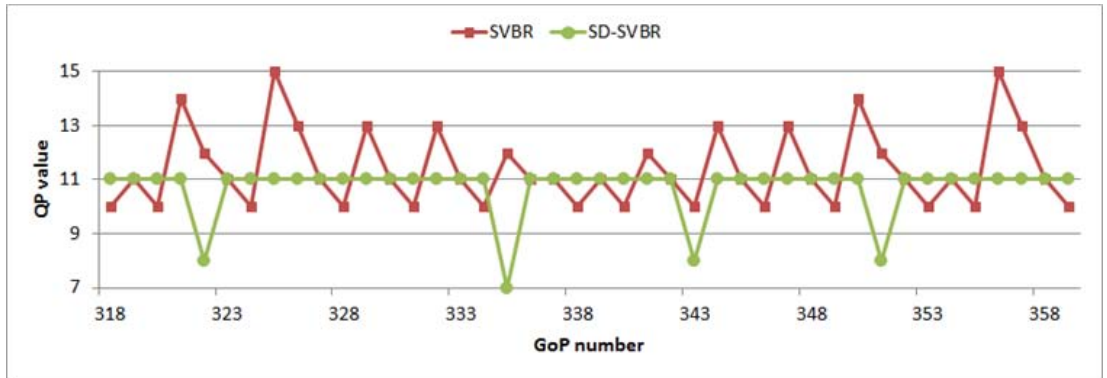
Sub-part 2 of Part IV chart video rate, as illustrated in Figure 6.23a, also exhibits a similar trend to sub-part 1, in terms of the PSNR value gained, which is not reflected in the corresponding gained video rate. In the PSNR chart (refer to Figure 6.20b), SD-SVBR gains a much higher value than SVBR, whereas in terms of the video rate, SVBR gains a slightly higher video rate than SD-SVBR (refer to Figure 6.23a).

However, the slightly higher SVBR video rate gained is also attained with high penalty of the bucket overflow. As discussed previously, the real bucket overflow might be different from what has been shown. Figure 6.24 shows the actual bucket utilization





(a) QP before GoPs 303-317



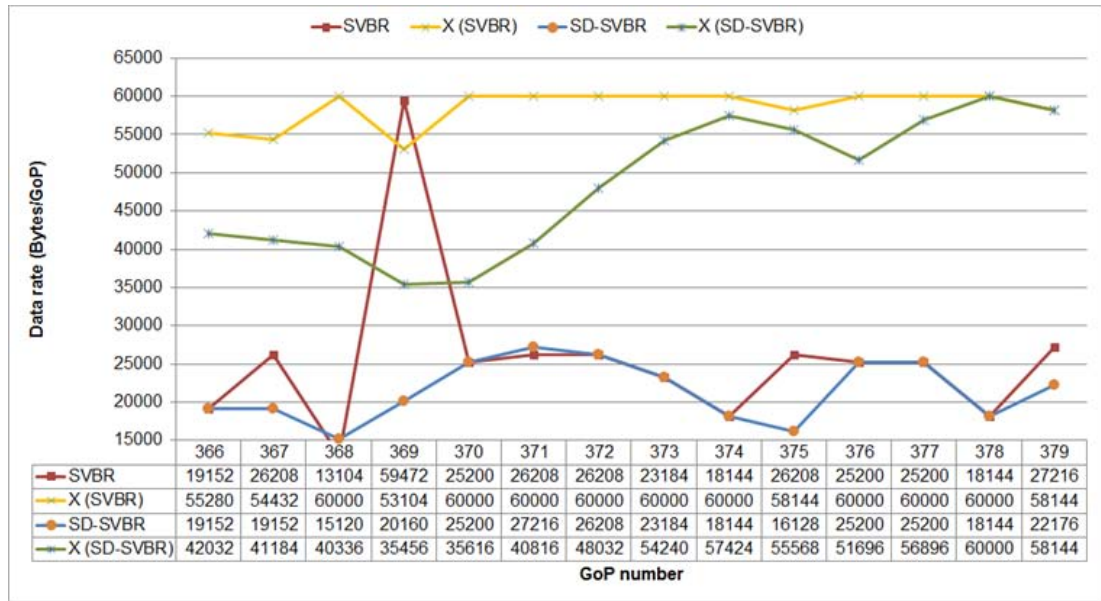
(b) QP after GoPs 303-317

**Figure 6.22:** QP before and after GoPs 303-317

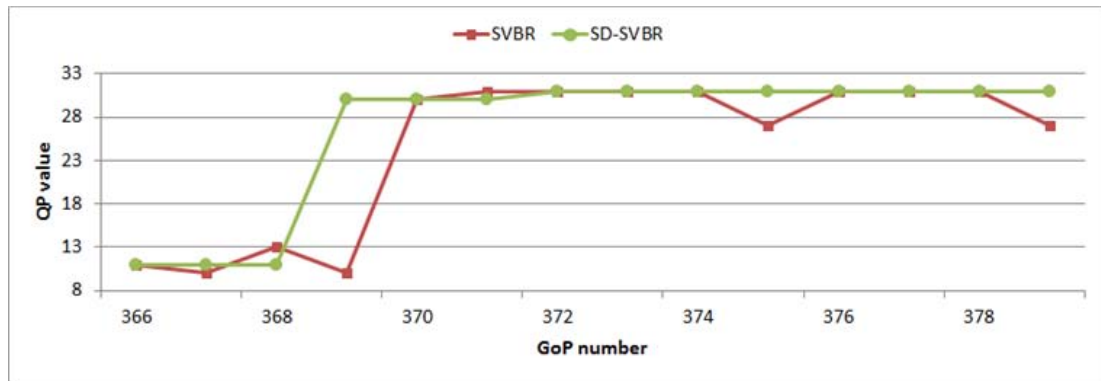
for both algorithms.

From the figure, it is clearly shown that the actual bucket utilization for the SVBR algorithm is very much overflowed, while, the bucket utilization for the SD-SVBR algorithm is in full utilization and there are only two GoPs which are slightly overflowed. This high overflow implication, especially for SVBR algorithm, is disastrous. The frame might be arriving in a long delayed time or it will be dropped as a result of congestion in the network.

Bucket overflow rarely occurs in SD-SVBR, as happened in the GoPs 378 and 379, but it is very difficult to be avoided completely. This is due to highly active video sequences that are processed at that time. Thus, it is very difficult to reduce the video rate to the an acceptable level. For instance, at GoP 377 the lowest video rate is 25200 Bytes/GoP by using  $QP = 31$ . At that time (refer to Figure 6.23a) the bucket utilization



(a) video rate for GoPs 366-379



(b) QP for GoPs 366-379

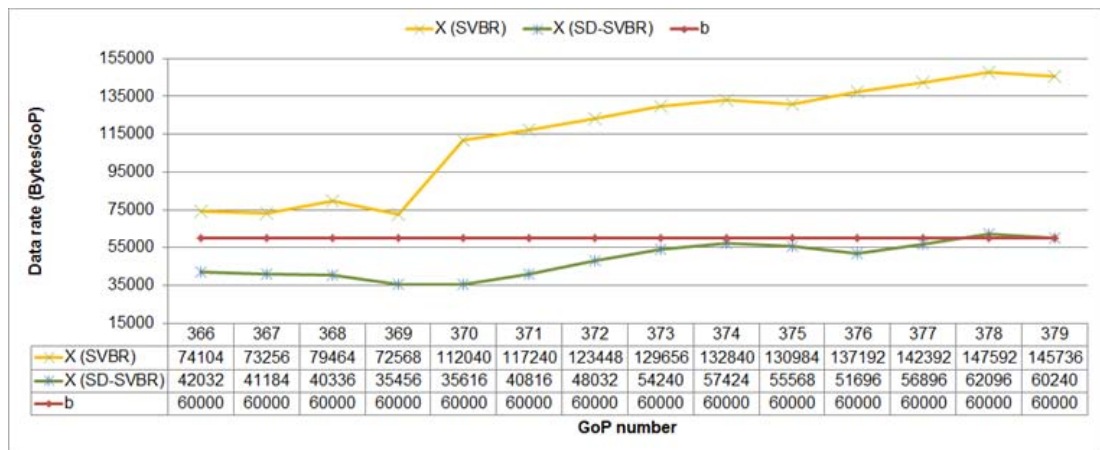
**Figure 6.23:** The video rate and QP value for sub-part 2 of Part IV chart

is already almost 57000. However, the SD-SVBR algorithm is still able to control its video rate so that the bucket overflow is at the very minimum. This is due to the strategy in reserving some spaces in the bucket to prepare for this kind of burtiness.

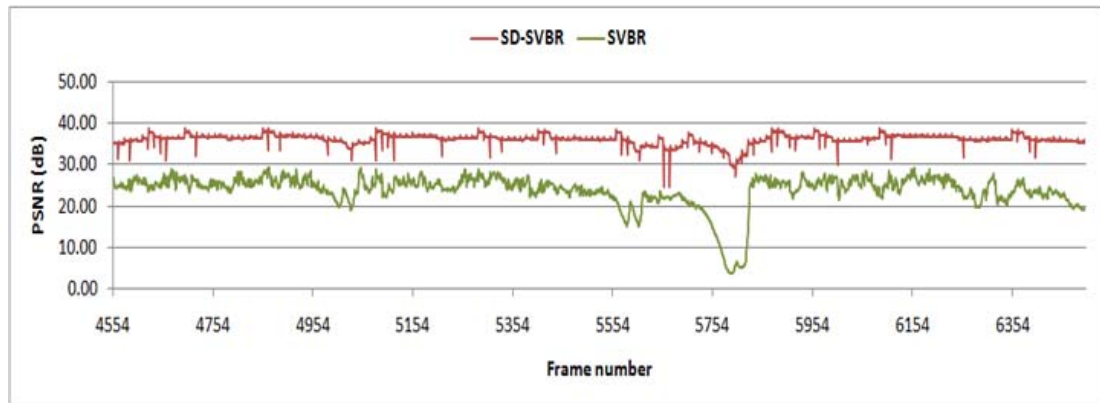
Besides the above-mentioned advantages gained by SD-SVBR, it also obtains smoother QP values as compared to SVBR (refer to Figure 6.23b).

## 6.2.5 Analysis: Part V of the PSNR Result

The Part V of the PSNR Result is depicted in Figure 6.25.



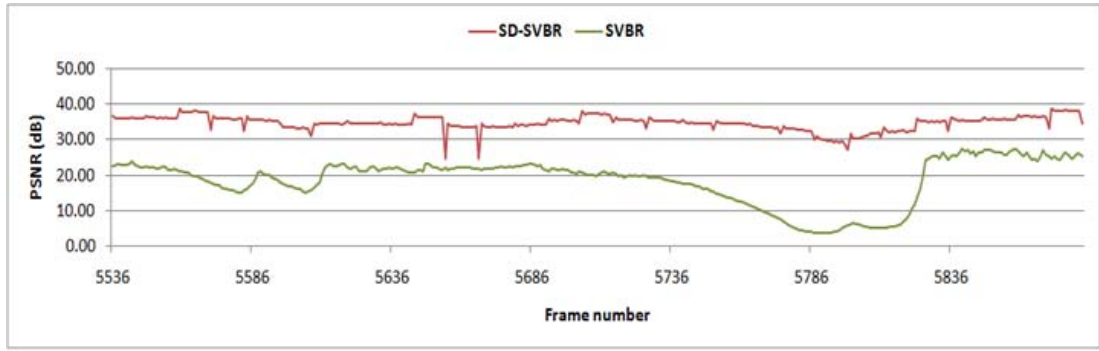
**Figure 6.24:** Actual “bucket” utilization for GoP 366-379



**Figure 6.25:** Part V: PSNR result for frames 4554-6499

To help in analyzing Part V of the PSNR chart (refer to Figure 6.25), only one sub-part is extracted from the chart. This is because the whole chart shows a similar chart trend. This sub-part is from frame 5536 to frame 5883. The equivalent GoPs are from 464 to 492. This is almost similar in trend from the previous subsection, where the difference in PSNR gained is more than 10db.

However, as in the previous subsection, the equivalent video rate for the video sequences does not reflect what the PSNR chart shows. The equivalent video rate and its bucket utilization are depicted in Figure 6.27. Both algorithms gain higher video rates inter-changeably. The bucket utilization by both algorithms is also at a high level, except at the beginning of the chart for SVBR algorithm (refer to Figure 6.27a).



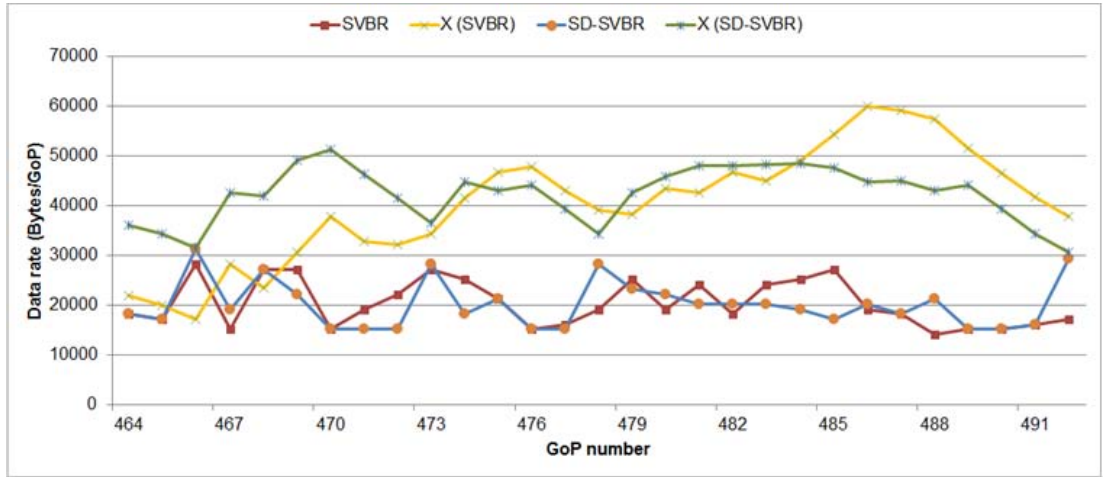
**Figure 6.26:** The PSNR sub-part for the Part V chart - frames

In terms of QP value gained by both algorithms, as displayed in Figure 6.27b, fluctuation is very obvious. Although, the fluctuation in SD-SVBR is slightly smoother than SVBR, but it is still considered high. This is the most difficult part to shape since the video sequences in this part are highly fluctuated. Without knowing the future video sequence, it is highly difficult to control in order to gain high video rate, while at the same time prevent overflow and be able to gain a stable QP value trend.

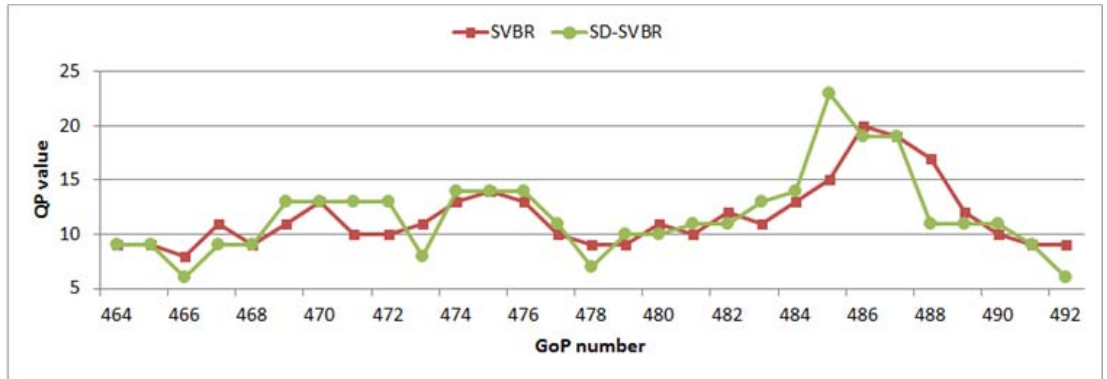
However, in shaping the video rate based on bucket utilization, the lesser utilized bucket at certain areas might be useful to be used immediately before or after those areas. For an algorithm with a high bucket utilization design, when the bucket is highly used at certain areas, it leaves a very small space for the future video sequence.

Therefore, although at the sub-part of Part V area, the SD-SVBR does not gain significant performance achievement in terms of higher video rate and smoother QP values, it has gained better performance before or after those areas. Figures 6.28 and 6.29 have charted a higher performance gained before and after those areas, either in the context of higher video rate, smoother QP values, or both.

Both before and after GoPs 464-492, SD-SVBR obtains an obvious smoother QP fluctuation than SVBR (refer to Figures 6.28b and 6.29b). In both charts, SVBR QP value is highly fluctuated. In addition, in terms of the video rate and bucket utilization, SD-SVBR gains better performance in both before and after GoPs 464-492 (refer to Figures 6.28a and 6.29a).



(a) video rate for GoPs 464-492

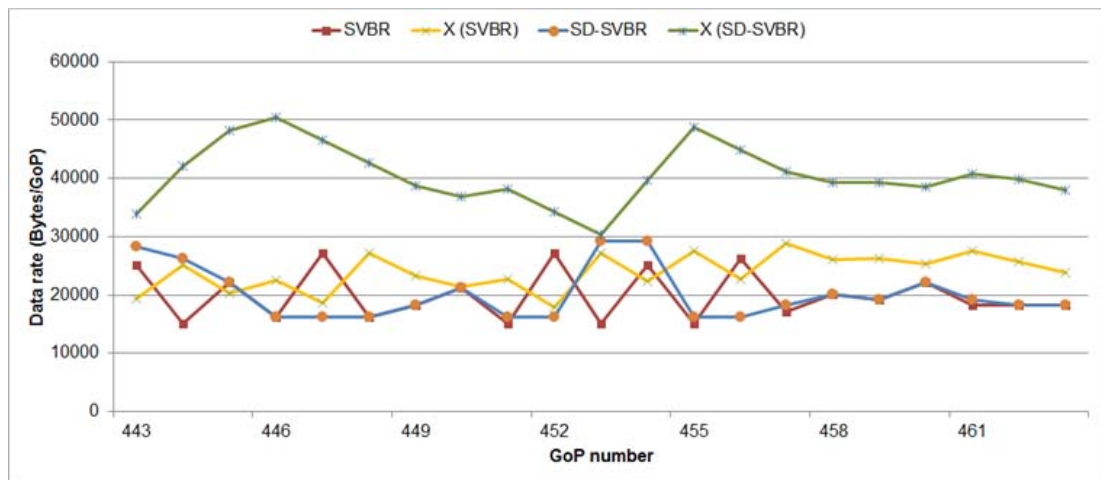


(b) QP for GoPs 464-492

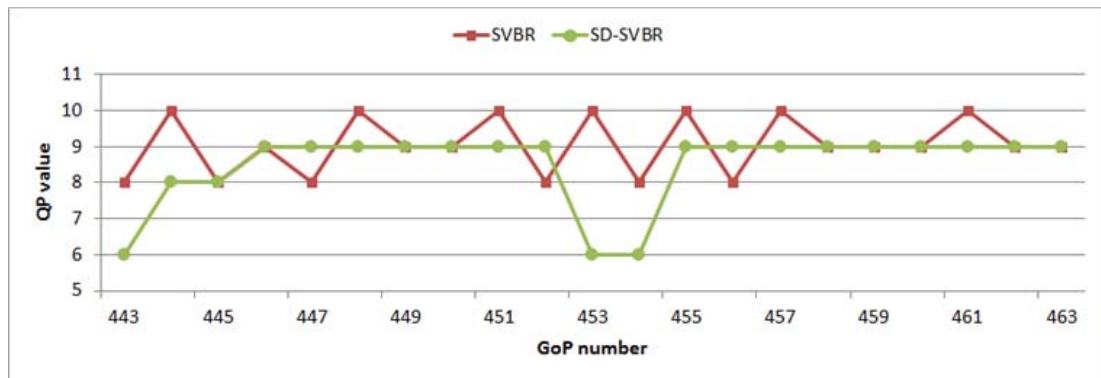
**Figure 6.27:** The video rate and QP value for sub-part of Part V chart

## 6.3 Summary

It is clear from the performance evaluation that by using various video sequence activities, that SD-SVBR has shown an impressive performance. It gains an impressive overall PSNR value. Besides that, in almost all cases, it gains a high video rate but without bucket overflow, it has utilized the bucket effectively well, and interestingly, it is still able to obtain smoother QP fluctuation. Although in some periods SD-SVBR did not gain higher PSNR value or higher bucket utilization, it has been shown that SD-SVBR is able to control the bucket utilization so that it is not overflowed or underflowed. In addition, it has been demonstrated that the SD-SVBR is able to gain good QP stability.



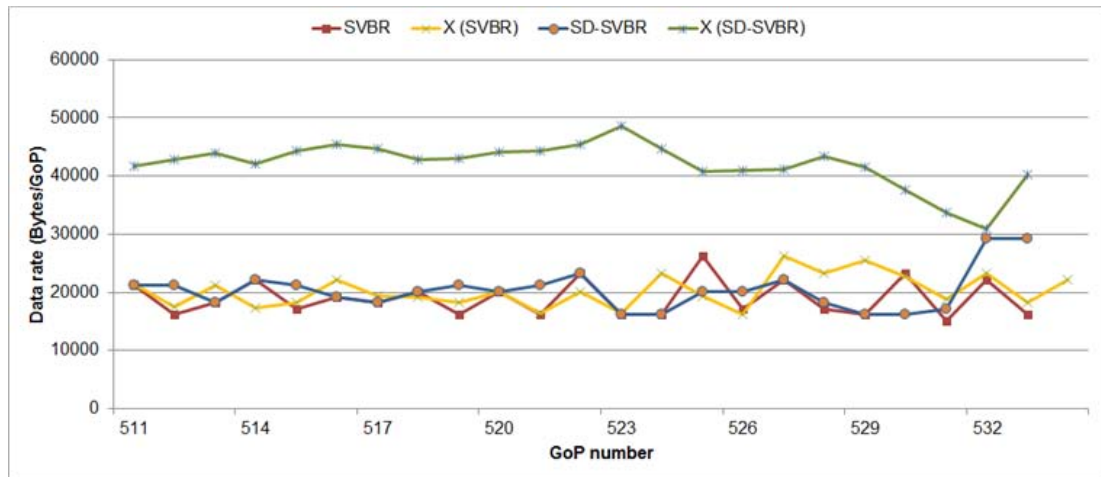
(a) video rate and bucket utilization at GoPs 443-463



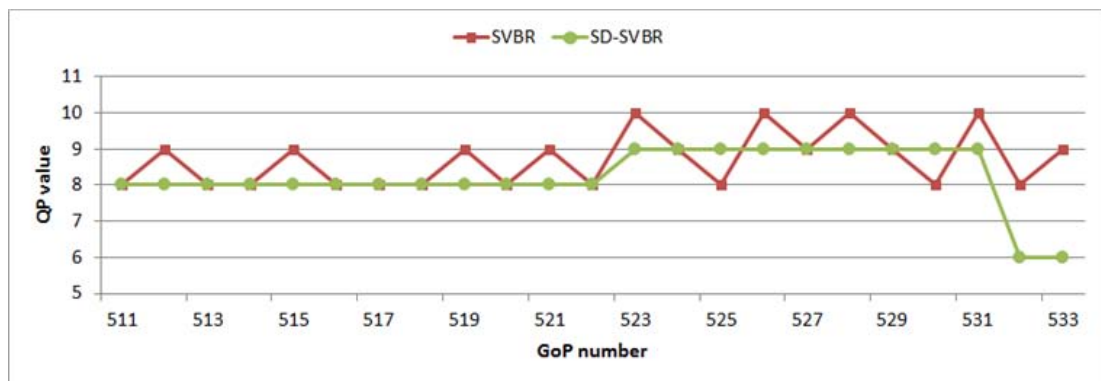
(b) QP values at GoPs 443-463

**Figure 6.28:** Higher performance gain before GoPs 464-492

In the next chapter, the global conclusions and contribution of the research work presented in this thesis will be stated. Then, some suggestions for further studies will be discussed.



(a) video rate and bucket utilization at GoPs 511-533



(b) QP values at GoPs 511-533

**Figure 6.29:** Higher performance gain after GoPs 464-492



# **CHAPTER SEVEN**

## **CONCLUSION AND FUTURE RESEARCH WORK**

After presenting the important aspects of this research, which were the problems of the previous rate shaping algorithm, the design of the new rate shaping algorithm, and its performance evaluation, this chapter will conclude the whole research work reported in this thesis and suggest future research work.

### **7.1 Summary of the Research**

As stressed throughout the thesis, the rising of video applications over the Internet has posed great challenges to the application or the Internet providers with regard to enable a smooth and high quality video application on the Internet. The Internet is initially designed for the traditional type of services, such as www, e-mail, telnet, ftp, and etc. Whereas, the video data is highly different from traditional type of applications in several ways; firstly, it is very large in size or transmission rate, secondly, it is highly bursty, and thirdly, it is sensitive to delay rather than error in the transmission.

The Internet transport protocol, particularly the two most dominant protocols nowadays, which are TCP and UDP, were created several decades ago. They are more suited for traditional types of applications. Although there is much active research on the aspect of creating or modifying the transport protocol to make it more suitable for real-time applications, more work in terms of real-time implementation is applicable



at the application layer. Among the most active work is on video coding and video rate control.

In the context of video rate control, it is of highly important work due to the fact that video sequences comprise widely varying content, motion, and have to arrive at the receiver at a certain acceptable time, while, the bandwidth of the Internet transmission channel is full of uncertainties, such as loss and delay of video traffic. As a consequence, in the last decade, an obvious growing in research on video rate control or video coding can be observed. The great challenge also can be seen on providing a smooth and a high video quality for real-time video transmission. This is true (many works in the real-time video application) by observing that there are lesser studies in the delay-based video application research, such as in Video on Demand applications.

Although the video rate control in real-time applications was initiated a couple of decades ago, they are still active nowadays, even at a multiple rate increment, since users are expecting a smoother and higher quality video playback. The novel Shaped VBR work by Hamdi et al. in [23] is one of the commendable works in this domain. They have proposed a real-time video rate control, which utilizes a leaky-bucket algorithm but without delay. This is due to the “bucket” (buffer) in the algorithm being more of an imaginary bucket, and the rate control is implemented “on the fly” without a buffer requirement. The SVBR algorithm is also relatively simple and performed at GoP video granularity which makes it further less subjected to overhead delay.

The detailed empirical analysis performed, as discussed in Chapter 4, has shown that this algorithm has demonstrated its novel creation of an ideal rate controller. In comparing with the traditional rate controllers, VBR and CBR, SVBR has manifested its advantages; it produces relatively a higher video rate, it is quick to adapt the video rate to within a permissible bandwidth level, and there is less delay as a result of no introduction of an additional buffer or complex computational algorithm.

However, there are spaces for performance improvement of the SVBR algorithm, especially under certain specific conditions. Besides the strengths of SVBR algorithm, the analysis had pointed several glaring weakness areas that can be considered for improvement:

- When an abrupt and sharp decrease or increase in the VBR video rate occurs, follow by an inverse sharp decrease or increase, the video rate in the SVBR algorithm will be increased or decreased “extremely”.
- The SVBR video rate tended to be at a low rate if the video rate in VBR and CBR are at the low rate as well. This condition persists even when the network bandwidth is idle or the leaky bucket is low occupied.
- The SVBR video rate is inclined to be in negative fluctuation, especially when there is an occurrence of high video rate video sequence.

Although there are many recent works, which based their studies on applying the SVBR algorithm as mentioned in Chapter 2, most of them were done under the limitation of the algorithm, or only performed minor changes to the core of the SVBR algorithm. Therefore (from the empirical analysis done), it clearly indicated that, a new shaped algorithm is needed, which have the following capabilities; no unwanted sharp video rate increment or decrement, a high video rate algorithm (without overflow and underflow as well), and a smoother QP stability.

All these have led to the creation of the SD-SVBR algorithm, which possesses the above-mentioned capabilities. Although, it has been named Slight Delay (SD) SVBR, it is just a delay in one GoP duration, which is less than half a second. By applying a one GoP delay, SD-SVBR has been creatively designed to gain a high video rate but without bucket overflow, to utilize the bucket well, and interestingly, it is still able to obtain a smoother QP fluctuation.

SD-SVBR algorithm has been designed with three main objectives, which are to remove unwanted video rate that is sharply increase or decrease as a consequence of the estimation/prediction used in the SVBR algorithm, to create a higher video rate algorithm, and to produce a smoother QP fluctuation. The results of SD-SVBR performance evaluation were shown in Chapter 6. The performance evaluation of SD-SVBR has been executed through carefully-employed experiments in all stages of the research methodology, as discussed in Chapter 3. It significantly outperforms SVBR algorithm in various aspects, particularly in producing a high video rate and at the same time better Quantization Parameter (QP) stability video sequence. Besides that, the video rate has been shaped efficiently to prevent unwanted sharp increments or decrements, and to avoid bucket overflow.

## **7.2 Contributions**

The important contributions of this research, as stated in Chapter 1 are as follows:

- The empirical evaluation of the SVBR algorithm, which has identified the strengths and weaknesses of the algorithm, as described in Subsection 7.2.1.
- A new shaped VBR algorithm for video application, as described in Subsection 7.2.2.
- The implementation of the SD-SVBR in a network simulator (ns-2), capable of working in different Group of Picture (GoP) sizes, and perform a user-perceived video performance evaluation, as described in Subsection 7.2.3.

### **7.2.1 Empirical Evaluation of the SVBR Algorithm**

By performing an extensive evaluation via an in-depth analysis and experiments of the SVBR algorithm, the detailed strengths and weaknesses of the algorithm have been

identified. These findings have contributed to the development of SD-SVBR algorithm and can be a guideline to others that intend to apply or modify the SVBR algorithm.

The findings found (on the strengths of the SVBR algorithm) that the SVBR algorithm has demonstrated its novel creation of an ideal rate controller, especially for a real time video application that produces a higher visual quality video, its video rate is within a permissible bandwidth level, and less delay, because of the non-existence of additional buffer or complex computational algorithm. These strengths, among other things, have been contributed by the usage of the leaky bucket algorithm, which is capable of controlling the video admission rate into a permissible bandwidth level. Although the overflow still exists, the usage of the leaky-bucket algorithm is able to quickly adjust the video rate to a permissible bandwidth level.

On the weaknesses of the algorithm, it was found that the design of the SVBR algorithm has contributed to the occurrence of an unwanted sharp decrease/increase in the VBR video rate, the occurrence of bucket overflow, the existence of a low video rate with a low bucket fullness level, and the generation of a cyclical negative fluctuation. The fundamental contributor for the above weaknesses has been identified as a consequence of the estimation and prediction used in generating the SVBR video rate. The existence of a low video rate with a low bucket fullness level is due to the SVBR principle that shapes the video rate to be lying between VBR and CBR video rates. The problem of the bucket overflow and a cyclical negative fluctuation are as a result of that SVBR algorithm that does not really dimension the video rate to the bucket fullness level and does not provide space (in the bucket) for smoothing the fluctuation.

### **7.2.2 A New Shaped VBR Algorithm**

As described in Chapter 5, one of the main challenges in designing the new video shaped algorithm is on generating a high video rate, but without bucket overflow

and minimum fluctuation. It has been discussed that the fundamental contributor for the weaknesses and limitations of the SVBR algorithm was due to the employment of estimation and prediction method in generating the SVBR video rate. Thus, the fundamental improvement is to avoid, as much as possible, the estimation and prediction in determining the bit rate allocation and its respective QP value.

However, as stressed in the chapter, it is not always possible to accomplish that principle in a real-time video application due to several reasons. Firstly, some forms of estimation or prediction have to be implemented since it is not always possible to obtain advanced information. Secondly, in particular for the leaky-bucket algorithm implementation, any allocation of the bit rates has implications. For instance, by allocating a high bit rate for the next video sequence, it leaves a very small space in the bucket. As a consequence, only small video rates can be allocated for the following video sequence. Thus, besides producing a low quality video sequence, it also contributes to the fluctuation in the video rates for the consecutive video sequences.

Thus, the research contribution is on the designs of video rate algorithm, which manipulates one GoP of video sequence in advance. By manipulating and implementing one GoP video data in advance, several interesting features have been embedded in the design. This new algorithm, which has been named SD-SVBR, has the following sub contributions:

- i. The algorithm can prevent unwanted sharp increments or decrements in the video rate.
- ii. The algorithm can produce a high video rate.
- iii. The algorithm can produce a smoother negative fluctuation.

These contributions are gained by processing the next GoP video sequence and obtaining the QP-to-video rate list. By obtaining the list of corresponding QP-to-video

rates, the rate controller does not only get a correct QP-to-video rate matching, but also a further opportunity to manipulate or shape the next video rate. For instance, the implication of using a certain QP value for the next GoP video rate and the bucket level, can be assessed beforehand. Thus, the rate control engine can choose a more appropriate QP value.

In producing a high video rate, a smart strategy has been proposed, which was by dimensioning the video rate to the higher utilization of the leaky-bucket, rather than based on CBR-VBR constraints as in SVBR. To increase the video rate, the intended video rate has to be increased and the corresponding QP value should be decreased. However, as highlighted in Chapter 5, the video rate cannot be simply increased in order to reduce the risk of bucket overflow and the fluctuation effect. Thus, an increment strategy based on scene activity video rate and the bucket level fullness had been devised. In addition, appropriate space has to be reserved in the bucket to reduce the video rate negative fluctuation.

In producing a smoother negative fluctuation, a carefully-measured of following the previous QP value and taking a middle value where the difference between previous QP and the intended QP value is wide, had been adopted. The strategy was to reserve some space in the bucket, so that, when the algorithm needs to follow the previous QP value, which might add some data in the bucket, it can be allowed.

### **7.2.3 The implementation of the SD-SVBR**

The SD-SVBR has been implemented in a strong popular-based network simulator, which is ns-2. By implementing the algorithm in a most-used simulation software, it indicates that the algorithm later on can be implemented in the real world.

## The Algorithm Implementation

This research has shown how to implement the coding which can perform the following features in the system:

- The designation of avoiding unwanted sharp increments or decrements in the video rate by obtaining a correct corresponding QP value. In short, when processing the GoP- $k$  video sequence, the rate control engine obtains, in advance, the GoP- $k+1$  information. Before generating the QP value for GoP- $k+1$ , the application processes the GoP- $k+1$  video sequence, and produces a list of corresponding QP-to-video rates. By obtaining the list, the intended video rate can be obtained via generating the appropriate QP value to be forwarded the encoder.
- The designation produces a high video rate by using an increment strategy, which is based on scene activity video rate and the bucket level fullness. Briefly, the rate will be increased as far as there is available space in the bucket, but appropriate space needs to be reserved in the bucket to reduce the negative fluctuation of the video rate. The buffer fullness level is divided into three categories according to the VBR video rate. These categories are as follows:
  - Category I: for the VBR video rate between 0 to  $y$  ( $\frac{r+(b/2)}{2}$ ), the bucket fullness level is targeted around 75% (3/4 bucket level).
  - Category II: for the VBR video rate between  $y$  and  $b/2$ , the bucket fullness level is targeted around 75% (3/4 bucket level).
  - Category III: for the VBR video rate above  $b/2$ , the bucket fullness level is targeted around 50% (1/2 bucket level).
- The designation of a smoother negative fluctuation by following the previous QP value and taking a middle value where the difference between previous QP and

the intended QP value is wide. In short, it follows the previous QP value if the following conditions are met, that it is still in a high video rate, the bucket is not overflowed, bucket is not under utilized, and there are still reserves in the bucket for future adjustment, to ensure that the three above conditions can still be met.

### **The Verification and Validation of the Implementation**

To ensure that the implementation is correct, the verification and validation processes for all related stages of the implementation have been executed. It involves examination of the simulation program to ensure that the operational model accurately reflects the conceptual model. The ns-2 has been validated using ns-2's collection of detailed validation scripts.

In designing the new algorithm, the design objectives were stated. Each objective has its own sub-algorithm, principle, or formulation. For each sub-algorithm, the verification was done by evaluating the result on either it conforms to the initial objective or formulation. All the sub-algorithms had been verified and the details of verification processes were discussed in Chapter 5.

The new algorithm implementation was validated on ns-2 by using Run-time Trace and Incremental Implementation. For every simulation, the Run-time Trace was checked to ensure it ran as expected. The validation was done by using multiple types of video sequence. The video test sequence is a combination of several video clips. The result showed that the whole the new algorithm has performed as it is intended, thus it can be concluded that the new algorithm has been validated.

### **Work Environment and Dynamic GoP Size**

The algorithm has been implemented within the Evalvid-RA environment, which provides tools or framework to evaluate the quality of video transmission either in a real or simulated network. Therefore, in implementing the algorithm, the researcher



had to go deeper inside the work environment, and then inserted the algorithm into various related libraries. Besides that, several changes needed to be done, in particular, to enable videos with different GoP sizes be implemented in the system. By using header file or H-frame, instead of using the consistent 12 frames as an indication of the new GoP, the system can adapt for whatever frame number per GoP.

## **7.3 Suggestions for Future Work**

The suggestions for future research work related to the video rate control are as follows:

- develop an integrated video rate control,
- exploit features in advanced codec, and
- analyze all video and optimize the video rate algorithm.

### **7.3.1 Integrated Rate Control**

As stated in Subsection 1.1.2, there are three schemes of rate control for video transmission, which are:

- controlling the video rate coding,
- regulating the transport protocol, and
- integrated control scheme.

This study is on the first scheme, which is controlling the video rate coding. In this scheme, the video data source is regulated so that in the network interface, there will be fewer bursts. Whereas, in the integrated rate control scheme, the rate controller gets feedback from the transport layer to produce better performance for the overall

transmission. As concluded in [175], any optimization on the video coding rate does not automatically enhance the overall video data transmission. This is due to the layering principle separation of the computer network architecture. The improvement of the video coding rate will forward generated data into the network interface. Then the transport layer protocol transmission will pick up the data for transmission. As such, improving the algorithm at the application layer will not necessarily improve the overall video data transmission.

As explained in [175], the Internet is built on the notion of protocol layering by breaking the complex task of network functions into self-regulating protocol layers. Each layer performs different operations with minimum interaction among them. In the context of this study, the main layers involved are the application layer and the transport layer. The application layer performs video coding rate control. Meanwhile, the transport layer regulates transmission rate in the Internet.

By referring to the video transmission rate architecture, as illustrated in Figure 1.1, the sender and the receiver perform application layer tasks. The video coding rate is done at the sender, while the processes (of the video coding rate) end at the network interface. Then the data will be sent to the receiver via the Internet. The Internet transmission rate for this study is regulated by TFRC transport protocol congestion control.

The layering principle has functioned well in the Internet in terms of scalability and functionality, at least for data applications. On the other hand, the layering makes it difficult to provide end-to-end performance guarantees. This is true in terms of fulfilling some performance requirements of the application, such as in the case of applications sensitive to delays. Thus, any optimization of video coding rates does not necessarily improve the video data transmission effectively.

However, one of the great challenges in performing the study under this scheme is in determining timing mismatch between video coding rate and TFRC transmission rate. This is due to the TFRC variabilities, in terms of Round-Trip Time (RTT), congestion state/packet loss event, and etc.

### **7.3.2 Advanced Codec Support**

This study implemented a video rate control for traditional MPEG IV, because the main purpose for the new algorithm is to create a new shaped VBR, which is targeted for a real-time application (although it is with a very slight delay). Thus, the new algorithm has to be simple in order to reduce the delay as much as possible, and it does not manipulate the new feature available in an advanced codec, for instance, H.264/MPEG-4 Part 10 or Advanced Video Coding (AVC).

Since the codec is already supported by the Evalvid in Evalvid 2.0, further study is required to utilize the advanced features in the codec. For example, in Scalable Video Coding (SVC), which can produce multiple layers of a video frame data, the algorithm might be able to manipulate spatial or temporal resolution video, to further control the video rate which is more adaptable to the variable rates of the network bandwidth.

By performing this support in the video rate control algorithm, it means that the application can adapt to a wide variety of video-capable hardware, for example, PDAs, cell phones, laptops, desktops, and the wired or wireless delivery networks. Each of these devices, which has different constraints due to processing power, viewing size, and so on, can have these different constraints satisfied with one encoding algorithm of the video.

However, the new algorithm should remain simple, in terms of the granularity or perspective of the video manipulation, to the level that the new algorithm does not produce excessive delay for a real-time video application. Thus, comparative studies need to be done to measure the delay, particularly the overhead delay, in manipulating

the available advanced features. Although theoretically it will generate delay, with advanced manipulation of the features it might alleviate the delay and produce a faster algorithm than the SD-SVBR.

### **7.3.3 Exploiting the Algorithm for Stored Video and/or Frame/Object Level**

As mentioned in Chapter 6, for instance in Subsection 6.2.2, there are still some constraints in the SD-SVBR in order to produce a more higher video rate or a more stable QP value. This is due to the fact that SD-SVBR exploits ahead information of only one GoP. By examining at several GoPs in advance, the algorithm can obtain more information on future video rates, thus it might be beneficial to exploit the information to the highest possible video rate and the optimum level of QP stability.

The granularity of the video data might be very useful in terms of designing a higher video rate and more stable QP value. With I-frame usually more multiple in data size or video rate than other types of video frame, it creates burstiness at that frame, thus the algorithm might be able to manipulate this scenario to reduce the burstiness in order to reduce the overall burstiness. The algorithm also might be able to use a frame discard feature when the network is not able to support more data, which might reduce the network congestion level, and it might lead to minimum effect of the frame lost (congested network will drop the packets arbitrarily).

By analyzing all GoPs of a video sequence, the algorithm also can be manipulated in producing a better video rate by exploiting the video object level. Each object in a video frame can be evaluated in order to determine the effect of the objects on the video rate. Therefore, based on network conditions, these objects can be manipulated in producing an optimum video quality.

# REFERENCES

- [1] R. Jain, *The Art of Computer System Performance Analysis*. John Wiley, 1991.
- [2] A. Lie and J. Klaue, “Evalvid-RA: Trace driven simulation of rate adaptive MPEG-4 VBR video,” *Multimedia Systems*, vol. 12, no. 1, 2008.
- [3] S. Hassan and M. Kara, “Simulation-based performance comparison of TCP-friendly congestion control protocols,” in *Proceedings of the 16<sup>th</sup> Annual UK Performance Engineering Workshop (UKPEW2000)*, Jul. 2000.
- [4] P. Seeling, M. Reisslein, and B. Kulapala, “Network performance evaluation using frame size and quality traces of single-layer and two-layer video: A tutorial,” *IEEE Communications Surveys Tutorials*, vol. 6, no. 3, pp. 58 –78, 2004.
- [5] L. Guo, E. Tan, S. Chen, Z. Xiao, O. Spatscheck, and X. Zhang, “Delving into internet streaming media delivery: A quality and resource utilization perspective,” in *Proceedings of the 6<sup>th</sup> ACM SIGCOMM conference on Internet measurement*. Rio de Janeiro, Brazil: ACM Press, 2006, pp. 217–230.
- [6] R. Zimmermann, C. Shahabi, and K. Fu, “A multi-threshold online smoothing technique for variable rate multimedia streams,” *Multimedia Tools and Applications*, vol. 28, no. 1, pp. 23–49, 2006.
- [7] A. Lie, “Enhancing rate adaptive IP streaming media performance with the use of Active Queue Management,” Ph.D. dissertation, Norwegian University of Science and Technology, 2008.
- [8] M.-J. Kim and M.-C. Hong, “Adaptive rate control scheme for real-time H.264/AVC video coding,” in *the Proceedings of 2010 Digest of Technical Papers International Conference on Consumer Electronics (ICCE)*, 2010, pp. 271–272.
- [9] M. Semsarzadeh, M. Langroodi, and M. Hashemi, “An adaptive rate control for faster bitrate shaping in x264 based video conferencing,” in *Proceedings of 2010 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, Mar. 2010, pp. 1 –4.

- [10] C.-W. Seo, J. W. Kang, J.-K. Han, and T. Nguyen, "Efficient bit allocation and rate control algorithms for hierarchical video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 9, pp. 1210 –1223, Sep. 2010.
- [11] P.-T. Wu, T.-C. Chang, C.-L. Su, and J.-I. Guo, "A BU-based rate control design for H.264 and AVS video coding with ROI support," in *Proceedings of 2010 International Symposium on VLSI Design Automation and Test (VLSI-DAT)*, Apr. 2010, pp. 215 –218.
- [12] X. Liu, Q. Dai, and C. Fu, "Improved methods for initializing R-Q model parameters and quantization parameter in H.264 rate control," in *Proceedings of 2009 WRI World Congress on Computer Science and Information Engineering*, vol. 6, Mar. 2009, pp. 320 –323.
- [13] T. I. Zhao Min and S. Goto, "A novel rate control algorithm for H.264/AVC," in *Proceedings of The 23<sup>rd</sup> International Technical Conference on Circuits/Systems, Computers and Communications*. The Institute of Electronics, Information and Communication Engineers (IEICE), 2008, pp. 725 – 728.
- [14] Z. Chen and K. N. Ngan, "Recent advances in rate control for video coding," *Image Commun.*, vol. 22, no. 1, pp. 19–38, 2007.
- [15] H. A. Marios C. Angelides, *The Handbook of MPEG Applications: Standards in Practice*. John Wiley and Sons, 2011.
- [16] S. Eshaghi and H. Farsi, "Rate control and mode decision jointly optimization in H.264AVC," in *Proceedings of the 4<sup>th</sup> conference on European computing*, ser. ECC'10. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2010, pp. 280–283. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1844367.1844414>
- [17] P. Ranganth and Y. V. Swaroop, "Adaptive rate control for H.264 video encoding in a video transmission system," in *Proceedings of XXXII National Systems Conference (NSC 2008)*, Dec. 2008.
- [18] W. Lu, X. Gao, Q. Deng, and T. Wang, "A basic-unit size based adaptive rate control algorithm," in *Proceedings of Fourth International Conference on Image and Graphics (ICIG 2007)*, Aug. 2007, pp. 268 –273.
- [19] M. Jiang, "Adaptive rate control for advanced video coding," Ph.D. thesis, Santa Clara University, 2006.
- [20] H. Wang, "Robust image and video coding with adaptive rate control," Ph.D. dissertation, University of Nebraska at Lincoln, 2009.

- [21] X. Li, N. Oertel, A. Hutter, and A. Kaup, "Rate-distortion optimized frame level rate control for H.264/AVC," in *Proceedings of 16<sup>th</sup> European Signal Processing Conference (EUSIPCO 2008)*, 2008.
- [22] X. Zhu, "Distributed rate allocation for video streaming over wireless network," Ph.D. dissertation, The Department of Electrical Engineering of Stanford University, 2009.
- [23] H. Hamdi, J. W. Roberts, and P. Rolin, "Rate control for VBR video coders in broad-band networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, pp. 1040–1051, 1997.
- [24] C.-H. Ke, C.-K. Shieh, W.-S. Hwang, and A. Ziviani, "An evaluation framework for more realistic simulations of MPEG video transmission," *Journal of information science and engineering*, vol. 24, pp. 425–440, 2008.
- [25] S. Hu, H. Wang, S. Kwong, and T. Zhao, "Frame level rate control for H.264/AVC with novel rate-quantization model," in *Proceedings of 2010 IEEE International Conference on Multimedia and Expo (ICME)*, Jul. 2010, pp. 226–231.
- [26] H. Hu, J. Yang, L. Zhu, and H. Xi, "A scene-aware adaptive media playout algorithm for wireless video streaming," in *2010 The 2<sup>nd</sup> International Conference on Computer and Automation Engineering (ICCAE)*, vol. 1, Feb. 2010, pp. 399–403.
- [27] W. Kim, J. You, and J. Jeong, "Complexity control strategy for real-time H.264/AVC encoder," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 2, pp. 1137–1143, May 2010.
- [28] Q. Lin and G. Feng, "The bit allocation and RDO mode based rate control algorithm," in *Proceedings of 2010 International Conference on Anti-Counterfeiting Security and Identification in Communication (ASID)*, Jul. 2010, pp. 154–157.
- [29] E. Brosh, S. A. Baset, V. Misra, D. Rubenstein, and H. Schulzrinne, "The delay-friendliness of TCP for real-time traffic," *IEEE/ACM Trans. Netw.*, vol. 18, no. 5, pp. 1478–1491, 2010.
- [30] D. Damjanovic and M. Welzl, "MulTFRC: providing weighted fairness for multimedia applications (and others too!)," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 3, pp. 5–12, 2009.
- [31] M. A. Talaat, M. A. Koutb, and H. S. Sorour, "PSNR Evaluation of Media Traffic over TFRC," *International Journal of Computer Networks & Communications (IJCNC)*, vol. 1, no. 3, pp. 71–76, 2009.

- [32] E. Tan, J. Chen, S. Ardon, and E. Lochin, "Video TFRC," in *Proceedings of IEEE International Conference on Communications (IEEE ICC 2008)*, Beijing, China, 2008.
- [33] Z. Wang, S. Banerjee, and S. Jamin, "Media-aware rate control," Department of EECS, University of Michigan, Tech. Rep. CSE-TR-501-04, Nov. 2004.
- [34] H. Feng, C. Yuan, and Y. Li, "Self-adaptive control of video transmission based on available bandwidth estimation," in *Proceedings of 2010 International Conference on Communications and Mobile Computing (CMC)*, vol. 1, Apr. 2010, pp. 361–364.
- [35] S. Hasan, L. Lefevre, Z. Huang, and P. Werstein, "Cross layer protocol support for live streaming media," in *Proceedings of Advanced Information Networking and Applications (AINA 2008)*, 2008, pp. 319–326.
- [36] C. H. Shih, C. K. Shieh, J. Y. Wang, and W. S. Hwang, "An integrated rate control scheme for TCP-friendly MPEG-4 video transmission," *Image Commun.*, vol. 23, no. 2, pp. 101–115, 2008.
- [37] C.-H. Ke, C.-K. Shieh, W.-S. Hwang, and A. Ziviani, "Improving video transmission on the Internet," *IEEE Potentials*, vol. 26, no. 1, pp. 16–19, Jan. 2007.
- [38] D. C. Steven Bauer and W. Lehr, "The evolution of Internet congestion," in *Proceedings of 37th Research Conference on Communication, Information and Internet Policy*, 2009.
- [39] K. Jack, *Video demystified: a handbook for the digital engineer*. Newnes Publication, 2007.
- [40] M. Kumar and S. Verma, "Querying video sensor networks," in *Proceedings of Third International Conference on Wireless Communication and Sensor Networks (WCSN '07)*, Dec. 2007, pp. 50–54.
- [41] P. Seeling and M. Reisslein, "Evaluating multimedia networking mechanisms using video traces," *IEEE Potentials*, vol. 24, no. 4, pp. 21–25, Oct. 2005.
- [42] H. Bidgoli, *The Internet encyclopedia, Volume 1*. John Wiley and Sons, 2004.
- [43] A. Nguyen and J. Hwang, "SPEM rate control," in *Advances in multimedia information processing-PCM 2001: second IEEE Pacific-Rim Conference on Multimedia (IEEE-PCM 2001)*, 2001.
- [44] P. Seeling, F. H. P. Fitzek, and M. Reisslein, *Video Traces for Network Performance Evaluation: A Comprehensive Overview and Guide on Video Traces and Their Utilization in Networking Research*. Springer, 2007.



- [45] H. Zhao, N. Ansari, and Y. Q. Shi, "Layered MPEG video transmission over IP DiffServ," in *International Conference on Information Technology: Coding and Computing*, vol. 1. Los Alamitos, CA, USA: IEEE Computer Society, 2005, pp. 63–67.
- [46] 4-H Filmmaking, "Introduction to video," 2009, [Online; accessed 26-Jan-2011]. [Online]. Available: <http://online.4-hcurriculum.org/>
- [47] Wikipedia, "Frame rate," 2011, [Online; accessed 26-Jan-2011]. [Online]. Available: [http://en.wikipedia.org/wiki/Frame\\_rate](http://en.wikipedia.org/wiki/Frame_rate)
- [48] —, "Common Intermediate Format," 2011, [Online; accessed 29-Jan-2011]. [Online]. Available: [http://en.wikipedia.org/wiki/Common\\_Intermediate\\_Format](http://en.wikipedia.org/wiki/Common_Intermediate_Format)
- [49] MSDN, "About YUV video," 2011, [Online; accessed 29-Jan-2011]. [Online]. Available: <http://msdn.microsoft.com/en-us/library/bb530104%28v%28vs.85%29.aspx>
- [50] Apple Inc., "MPEG-2 Reference Information," 2010, [Online; accessed 29-Jan-2011]. [Online]. Available: <http://documentation.apple.com/en/compressor/>
- [51] A. S. M. Arif, S. Hassan, O. Ghazali, and M. M. Kadhum, "Enhancing Shaped VBR rate control algorithm for stored video transmission system," in *Proceeding of The 2010 International Conference on Modeling, Simulation and Control (ICMSC 2010)*, 2010.
- [52] H. Sun, A. Vetro, and J. Xin, "An overview of scalable video streaming: Research articles," *Wireless Communications & Mobile Computing*, vol. 7, no. 2, pp. 159–172, 2007.
- [53] H. d. G. Carolina Blanch and P. van der Stok, "Inventory of MPEG-4 codecs," Information Society Technologies, Tech. Rep. BETSY IST-2004-004042, 2005.
- [54] Wikipedia, "Inter frame," 2011, [Online; accessed 29-Jan-2011]. [Online]. Available: [http://en.wikipedia.org/wiki/Inter\\_frame](http://en.wikipedia.org/wiki/Inter_frame)
- [55] J.-R. Ohm, "Overview of scalable video coding," International Organisation for Standardisation - ISO/IEC JTC1/SC29/WG11, Tech. Rep., 2008. [Online]. Available: <http://mpeg.chiariglione.org/technologies/mpeg-4/mp04-svc/index.htm>
- [56] P. Topiwala, "Introduction and overview of Scalable Video Coding (SVC)," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, ser. Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference, vol. 6312, Sep. 2006.

- [57] F. Frank, S. Patrick, and R. Martin, "Video streaming in wireless internet," in *Mobile Internet*, ser. Electrical Engineering & Applied Signal Processing Series. CRC Press, Apr. 2004. [Online]. Available: <http://dx.doi.org/10.1201/9780203499986.ch11>
- [58] W. Li, "Overview of Fine Granularity Scalability in MPEG-4 video standard," *IEEE Transactions on*, vol. 11, pp. 301–317, 2001.
- [59] P. de Cuetos, K. W. Ross, and M. Reisslein, "Evaluating the streaming of FGS-encoded video with rate-distortion traces," Institut Eurecom, Tech. Rep. RR-03-078, Jun. 2003.
- [60] S. L. Yao Wang, A. R. Reibman, "Multiple description coding for video delivery," in *Proceedings of the IEEE*, vol. 93, no. 1, 2005, pp. 57–70.
- [61] T. Y. Tian, "Research and applications of rate control in video coding," *Journal of University of Electronic Science and Technology of China*, pp. 24–32, 2006.
- [62] Z. Zhu, Y. Bai, Z. Duan, and F. Liang, "Novel rate-control algorithm based on TM5 framework," *Wireless Sensor Network*, vol. 1, pp. 182–188, 2009.
- [63] N. Eiamjumrus and S. Aramvith, "Rate control scheme based on cauchy R-D optimization model for H.264/AVC under low delay constraint," in *Proceedings of International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP '06)*, Dec. 2006, pp. 205–210.
- [64] L. Chiariglione, "Short MPEG-2 description," International Organisation for Standardisation - ISO/IEC JTC1/SC29/WG11, Tech. Rep., 2000. [Online]. Available: <http://mpeg.chiariglione.org/standards/mpeg-2/mpeg-2.htm>
- [65] Data-Compression.com, "Theory of data compression," 2010, [Online; accessed 29-Jan-2011]. [Online]. Available: <http://www.data-compression.com/theory.html>
- [66] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE Nat. Conventional Record, Part 4*, pp. 142–163, 1959.
- [67] C. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.
- [68] Wikipedia, "Information theory," 2011, [Online; accessed 29-Jan-2011]. [Online]. Available: [http://en.wikipedia.org/wiki/Information\\_theory](http://en.wikipedia.org/wiki/Information_theory)
- [69] A. S. M. Arif, S. Hassan, O. Ghazali, and S. A. Nor, "Evalvid-RASV: Shaped VBR rate adaptation stored video system," in *Proceedings of 2010 2<sup>nd</sup> International Conference on Education Technology and Computer (ICETC)*, vol. 5, Jun. 2010, pp. V5–246–V5–250.

- [70] S. Gringeri, K. Shuaib, R. Egorov, A. Lewis, B. Khasnabish, and B. Basch, "Traffic shaping, bandwidth allocation, and quality assessment for MPEG video distribution over broadband networks," *IEEE Network*, vol. 12, no. 6, pp. 94–107, Dec. 1998.
- [71] A. S. M. Arif, S. Hassan, O. Ghazali, and M. M. Kadhum, "VBR vs CBR: The Shaped VBR stored video SD-SVBR mechanism is the winner!" in *Proceedings of the 5<sup>th</sup> Social Economic and Information Technology (SEiT) conference*, 2010.
- [72] J.-B. Cheng and H.-M. Hang, "Adaptive piecewise linear bits estimation model for MPEG based video coding," in *Proceedings of International Conference on Image Processing*, vol. 2, Oct. 1995, pp. 551–554.
- [73] J.-J. Chen and H.-M. Hang, "Source model for transform video coder and its application. II. Variable frame rate coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 2, pp. 299–311, Apr. 1997.
- [74] H.-M. Hang and J.-J. Chen, "Source model for transform video coder and its application. I. Fundamental theory," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 2, pp. 287–298, Apr. 1997.
- [75] Z. He, Y. K. Kim, and M. S.K., "Low-delay rate control for DCT video coding via rho-domain source modeling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 8, pp. 928–940, Aug. 2001.
- [76] B. Xie and W. Zeng, "A sequence-based rate control framework for consistent quality real-time video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 1, pp. 56–71, Jan. 2006.
- [77] Z. Chen, J. Han, and K. N. Ngan, "A unified framework of unsupervised subjective optimized bit allocation for multiple video object coding," in *Proceedings of Multimedia Systems and Applications VIII*, vol. 6015, 2005, pp. 20–31.
- [78] S. Lee, M. Pattichis, and A. Bovik, "Foveated video compression with optimal rate control," *IEEE Transactions on Image Processing*, vol. 10, no. 7, pp. 977–992, Jul. 2001.
- [79] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998.
- [80] C.-W. Tang, C.-H. Chen, Y.-H. Yu, and C.-J. Tsai, "Visual sensitivity guided bit allocation for video coding," *IEEE Transactions on Multimedia*, vol. 8, no. 1, pp. 11–18, Feb. 2006.

- [81] H. Song and C.-C. Kuo, "Rate control for low-bit-rate video via variable-encoding frame rates," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 4, pp. 512–521, Apr. 2001.
- [82] A. S. M. Arif, O. Ghazali, and S. Hassan, "A survey on buffer and rate adaptation optimization in TCP-based streaming media studies," in *Proceedings of International Conference on Network Applications, Protocols and Services 2008 (NetApps2008)*, 2008.
- [83] B. Wang, W. Wei, J. Kurose, D. Towsley, K. R. Pattipati, Z. Guo, and Z. Peng, "Application-layer multipath data transfer via TCP: Schemes and performance tradeoffs," *Perform. Eval.*, vol. 64, no. 9-12, pp. 965–977, 2007.
- [84] A. Argyriou, "Real-time and rate-distortion optimized video streaming with TCP," *Image Commun.*, vol. 22, no. 4, pp. 374–388, 2007.
- [85] S. Boyden, A. Mahanti, and C. Williamson, "TCP vegas performance with streaming media," in *Performance, Computing, and Communications Conference (IPCCC 2007)*, 2007, pp. 35–44.
- [86] B. Wang, J. Kurose, P. Shenoy, and D. Towsley, "Multimedia streaming via TCP: an analytic performance study," *SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 1, pp. 406–407, 2004.
- [87] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Trans. Netw.*, vol. 7, no. 4, pp. 458–472, 1999.
- [88] R. Rejaie, M. Handely, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet," in *Proceedings of IEEE Infocom*, 1999.
- [89] M. Prangl, I. Kofler, and H. Hellwagner, "Towards QoS improvements of TCP-based media delivery," in *Proceedings of Networking and Services (ICNS 2008)*, 2008, pp. 188–193.
- [90] T. Kim and M. H. Ammar, "Receiver buffer requirement for video streaming over TCP," *Silicon.com Whitepaper*, 2005.
- [91] P. Mehra, "Efficient video streaming using TCP," University of California, Berkeley, EE228A Fall 2002 Final Project, 2002.
- [92] G. Ashvin, K. Charles, L. Kang, and W. Jonathan, *Supporting Low Latency TCP-Based Media Streams*. Oregon Graduate Institute School of Science & Engineering, 2002.
- [93] C. Krasic, K. Li, and J. Walpole, "The case for streaming multimedia with TCP," in *Proceedings of the 8<sup>th</sup> International Workshop on Interactive Distributed Multimedia Systems*. Springer-Verlag, 2001, pp. 213–218.

- [94] T. Phelan, "Strategies for streaming media applications using TCP-friendly rate control," Internet Engineering Task Force (IETF), Tech. Rep.
- [95] L. Guo, S. Chen, Z. Xiao, and X. Zhang, "Analysis of multimedia workloads with implications for internet streaming," in *Proceedings of the 14<sup>th</sup> international conference on World Wide Web*. Chiba, Japan: ACM, 2005, pp. 519–528.
- [96] M. Li, M. Claypool, R. Kinicki, and J. Nichols, "Characteristics of streaming media stored on the web," *ACM Trans. Inter. Tech.*, vol. 5, no. 4, pp. 601–626, 2005.
- [97] C. Krasic, "A framework for quality-adaptive media streaming: Encode once - stream anywhere," Ph.D. dissertation, Oregon Health & Science University, 2004.
- [98] J. G. Apostolopoulos, W.-T. Tan, and S. J. Wee, "Video streaming: Concepts, algorithms, and systems," in *Handbook of Video Databases, Design and Applications*. CRC Press, Sep. 2004.
- [99] J. Widmer, R. Denda, and M. Mauve, "A survey on TCP-friendly congestion control," *IEEE Network*, vol. 15, no. 3, pp. 28–37, 2001.
- [100] H. Elaarag and M. Bassiouni, "An Internet friendly transport protocol for continuous media over best effort networks," *International Journal of Communication Systems*, vol. 15, pp. 881–898, 2002.
- [101] D. Sisalem and A. Wolisz, "LDA+: Enhanced loss-delay based adaptation algorithm," in *Multimedia and Expo (ICME 2000)*, New York City, New York, USA., 2000.
- [102] I. Rhee, V. Ozdemir, and Y. Yi, "TEAR: TCP emulation at receivers - flow control for multimedia streaming," Tech. Rep., 2000.
- [103] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification," Internet Engineering Task Force, RFC 5348, Sep. 2008. [Online]. Available: <http://www.rfc-editor.org/rfc/rfc5348.txt>
- [104] E. Kohler, M. Handley, and S. Floyd, "Datagram Congestion Control Protocol (DCCP)," Internet Engineering Task Force, RFC 4340, Mar. 2006. [Online]. Available: <http://www.rfc-editor.org/rfc/rfc4340.txt>
- [105] K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," Internet Engineering Task Force, RFC 3168, Sep. 2001. [Online]. Available: <http://www.rfc-editor.org/rfc/rfc3168.txt>
- [106] Wikipedia, "Traffic shaping," 2011, [Online; accessed 31-Jan-2011]. [Online]. Available: [http://en.wikipedia.org/wiki/Traffic\\_shaping](http://en.wikipedia.org/wiki/Traffic_shaping)

- [107] Y. Wang and M. Jurczyk, "Impact of traffic shaping in ATM networks on video quality," in *Proceedings of 2000 International Workshops on Parallel Processing*, 2000, pp. 485–492.
- [108] N. Celandroni, E. Ferro, F. Potort, A. Chimienti, and M. Lucenteforte, "Dynamic rate shaping on MPEG-2 video streams for bandwidth saving on a faded satellite channel," *European Transactions on Telecommunications*, vol. 11, pp. 363–372, 2000. [Online]. Available: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=?doi=10.1.1.23.6403>
- [109] Wikipedia, "Leaky bucket," 2011, [Online; accessed 23-Jan-2011]. [Online]. Available: [http://en.wikipedia.org/wiki/Leaky\\_bucket](http://en.wikipedia.org/wiki/Leaky_bucket)
- [110] N. Yamanaka, Y. Sato, and K. Sato, "Performance limitation of the leaky bucket algorithm for ATM networks," *IEEE Transactions on Communications*, vol. 43, no. 8, pp. 2298–2300, Aug. 1995.
- [111] M. Alam, M. Atiquzzaman, and M. Karim, "Efficient MPEG video traffic shaping for the next generation internet," in *Proceedings of Global Telecommunications Conference (GLOBECOM '99)*, vol. 1A, 1999, pp. 364–368.
- [112] ———, "Effects of source traffic shaping on MPEG video transmission over next generation IP networks," in *Eight International Conference on Computer Communications and Networks*, 1999, pp. 514–519.
- [113] H. B. Kazemian, "An intelligent video streaming technique in zigbee wireless," in *Proceedings of the 18th international conference on Fuzzy Systems*, ser. FUZZ-IEEE'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 121–126. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1717561.1717583>
- [114] J. Klaue, B. Rathke, and A. Wolisz, "Evalvid - a framework for video transmission and quality evaluation," in *Proceedings of the 13<sup>th</sup> International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, 2003, pp. 255–272.
- [115] C. Bouras, V. Papapanagiotou, K. Stamos, and G. Zaoudis, "Efficient power management adaptation for video transmission over TFRC," in *Proceedings of 2010 Sixth Advanced International Conference on Telecommunications (AICT)*, May 2010, pp. 509–514.
- [116] C. Bouras, A. Gkamas, and G. Kioumourtzis, "Adaptive Smooth Simulcast Protocol (ASSP) for video applications: Description and performance evaluation," *Journal of Network and Systems Management*, pp. 1–35, 2010, 10.1007/s10922-010-9159-8. [Online]. Available: <http://dx.doi.org/10.1007/s10922-010-9159-8>

- [117] C. Bouras, G. Kioumourtzis, and A. Gkamas, "Performance evaluation of MPEG-4 video transmission with the Adaptive Smooth Multicast Protocol (ASMP)," in *Proceedings of 2010 IEEE Symposium on Computers and Communications (ISCC)*, Jun. 2010, pp. 540–545.
- [118] C. Bouras, A. Gkamas, and G. Kioumourtzis, "Evaluation of single rate multicast congestion control schemes for MPEG-4 video transmission," in *Proceedings of the 5<sup>th</sup> Euro-NGI conference on Next Generation Internet networks*, ser. NGI'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 32–39. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1671421.1671426>
- [119] D. Miras, "On quality aware adaptation of internet video," Ph.D. dissertation, Department of Computer Science, University College London, 2004.
- [120] A. C. Dalal and E. Perry, "A new architecture for measuring and assessing streaming media quality," in *Workshop on Passive and Active Measurement (PAM 2003)*, La Jolla, California, 2003.
- [121] Wikipedia, "Video quality," 2010, [Online; accessed 31-Jan-2011]. [Online]. Available: [http://en.wikipedia.org/wiki/Video\\_quality](http://en.wikipedia.org/wiki/Video_quality)
- [122] H. Koumaras, A. Kourtis, D. Martakos, and J. Lauterjung, "Quantified PQoS assessment based on fast estimation of the spatial and temporal activity level," *Multimedia Tools and Applications*, vol. 34, pp. 355–374, 2007, 10.1007/s11042-007-0111-1. [Online]. Available: <http://dx.doi.org/10.1007/s11042-007-0111-1>
- [123] M. S. Koul, "Error Concealment and Performance Evaluation of H.264/AVC Video Streams in a Lossy Wireless Environment," Dept. of Electrical Engineering, The University of Texas at Arlington, Tech. Rep., 2008.
- [124] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electronics Letters*, vol. 44, no. 13, pp. 800–801, 2008.
- [125] Wikipedia, "Peak signal-to-noise ratio," 2011, [Online; accessed 23-Jan-2011]. [Online]. Available: [http://en.wikipedia.org/wiki/Peak\\_signal-to-noise\\_ratio](http://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio)
- [126] WordIQ.com, "PSNR - definition," 2011, [Online; accessed 23-Jan-2011]. [Online]. Available: <http://www.wordiq.com/definition/PSNR>
- [127] O. Ghazali, "Scaleable and smooth TCP-friendly receiver-based layered multicast protocol," Ph.D. thesis, Universiti Utara Malaysia, 2008.
- [128] M. M. Kadhum, "Fast congestion notification mechanism for next generation routers," Ph.D. thesis, Universiti Utara Malaysia, 2010.
- [129] S. Floyd and V. Paxson, "Difficulties in simulating the Internet," *IEEE/ACM Transactions on Networking*, vol. 9, no. 4, pp. 392–403, 2001.

- [130] L. Qiu, Y. Zhang, and S. Keshav, "Understanding the performance of many TCP flows," Tech. Rep., 2001. [Online]. Available: <http://www.cs.utexas.edu/~yzhang/papers/manytcp-comnet01.pdf>
- [131] P. Barford and L. Landweber, "Bench-style network research in an internet instance laboratory," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 3, pp. 21–26, Jul. 2003. [Online]. Available: <http://doi.acm.org/10.1145/956993.956997>
- [132] O. M. D. Al-Momani, "Dynamic redundancy forward error correction mechanism for the enhancement of Internet-based video streaming," Ph.D. thesis, Universiti Utara Malaysia, 2010.
- [133] C. Williamson, "Internet traffic measurement," *IEEE Internet Computing*, vol. 5, no. 6, pp. 70–74, Dec. 2001.
- [134] D. Thomas, A. Joiner, W. Lin, M. Lowry, and T. Pressburger, "The unique aspects of simulation verification and validation," in *Proceeding of 2010 IEEE Aerospace Conference*, Mar. 2010, pp. 1–7.
- [135] O. B. Lynn, "A hybrid mechanism for SIP over IPv6 macromobility and micromobility management protocols," Ph.D. thesis, Universiti Utara Malaysia, 2008.
- [136] P. Hurtig, "Fast retransmit inhibitions for TCP," Master's thesis, Department of Computer Science, Karlstad University, 2006.
- [137] M. A. Law and K. W. David, *Simulation Modeling and Analysis*, 3rd ed. McGraw-Hill, Inc., 2000.
- [138] R. F. Sari, "Performance evaluation of active network-based unicast and multicast protocols," Ph.D. thesis, University of Leeds, UK, 2004.
- [139] S. Keshav, "Congestion control in computer networks," Ph.D. dissertation, University of California, 1991.
- [140] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. Mccanne, K. Varadhan, Y. Xu, and H. Yu, "Advances in network simulation," *Computer*, vol. 33, no. 5, pp. 59–67, May 2000. [Online]. Available: <http://portal.acm.org/citation.cfm?id=621475>
- [141] S. Bajaj, L. Breslau, D. Estrin, K. Fall, S. Floyd, P. Haldar, M. Handley, A. Helmy, J. Heidemann, P. Huang, S. Kumar, S. McCanne, R. Rejaie, P. Sharma, K. Varadhan, Y. Xu, and a. Z. Haobo Yu, "Improving simulation for network research," USC Computer Science Department, Tech. Rep. Technical Report 99-702, 1999.
- [142] W. D. Kelton, R. P. Sadowski, and D. T. Sturrock, *Simulation with Arena*. McGraw-Hill, 2004.



- [143] J.-J. P. Damien Magoni, "Influence of network topology on protocol simulation," in *ICN'01 - International Conference on Networking*, ser. Lecture Notes in Computer Science, vol. 2093, Colmar, France, 2001, pp. 762–770.
- [144] K. Pawlikowski, H. J. Jeong, and J. R. Lee, "On credibility of simulation studies of telecommunication network," *IEEE Communications Magazine*, vol. 40, no. 1, pp. 132–139, 2002.
- [145] M. S. Lane, A. H. Mansour, and J. L. Harpell, "Operations research techniques: A longitudinal update 1973-1988," *Interfaces*, vol. 23, pp. 63–68, 1993.
- [146] S. Floyd and E. Kohler, "Internet research needs better models," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 29–34, Jan. 2003. [Online]. Available: <http://doi.acm.org/10.1145/774763.774767>
- [147] C. Cairano-Gilfedder and R. Clegg, "A decade of Internet research advances in models and practices," *BT Technology Journal*, vol. 23, no. 4, pp. 115–128, Oct. 2005. [Online]. Available: <http://dx.doi.org/10.1007/s10550-006-0013-1>
- [148] X. Chen and F. Lu, "An improved frame layer rate control algorithm for H.264," *Communications in Computer and Information Science*, vol. 135, pp. 417–421, 2011.
- [149] R. Lu, J.-W. Lin, and T.-D. Chiueh, "Cross-layer optimization for wireless streaming via adaptive MIMO OFDM," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, Jun. 2010, pp. 2255–2258.
- [150] M. Tiwari, T. Groves, and P. Cosman, "Competitive equilibrium bitrate allocation for multiple video streams," *IEEE Transactions on Image Processing*, vol. 19, no. 4, pp. 1009–1021, Apr. 2010.
- [151] H. Mansour, P. Nasiopoulos, and V. Krishnamurthy, "Rate and distortion modeling of CGS coded scalable video content," *IEEE Transactions on Multimedia*, vol. PP, no. 99, p. 1, 2010.
- [152] Z. Cui and X. Zhu, "A low complexity MB layer rate control algorithm based on motion similarity for H.264," in *2010 International Conference on Wireless Communications and Signal Processing (WCSP)*, Oct. 2010, pp. 1–4.
- [153] L. Tian, Y. Sun, Y. Zhou, and X. Xu, "Analysis of quadratic R-D model in H.264/AVC video coding," in *2010 17<sup>th</sup> IEEE International Conference on Image Processing (ICIP)*, Sep. 2010, pp. 2853–2856.
- [154] H. Li, S. Xiao, C. Wu, and Y. Feng, "A pixel-level rate control algorithm for the optimal prediction residual," in *2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, Oct. 2010, pp. 603–606.

- [155] J. Koo and K. Chung, "MARC: Adaptive rate control scheme for improving the QoE of streaming services in mobile broadband networks," in *2010 International Symposium on Communications and Information Technologies (ISCIT)*, Oct. 2010, pp. 105 –110.
- [156] M. Hrarti, H. Saadane, M.-C. Larabi, A. Tamtaoui, and D. Aboutajdine, "A New approach of Rate-Quantization modeling for Intra and Inter frames in H.264 rate control," in *2009 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, Nov. 2009, pp. 474 –479.
- [157] Y. Chen, "An intra-rate estimation method for H.264 rate control," in *2<sup>nd</sup> International Congress on Image and Signal Processing (CISP '09)*, 2009, pp. 1 –3.
- [158] M.-J. Kim and M.-C. Hong, "Adaptive real-time rate control for H.264," in *Proceeding of 15<sup>th</sup> Asia-Pacific Conference on Communications (APCC 2009)*, Oct. 2009, pp. 265 –268.
- [159] S.-H. Kim, S. Kim, and J.-W. Suh, "An efficient rate control with adaptive quantization parameter decision and header bits length estimation," in *Digest of Technical Papers International Conference on Consumer Electronics (ICCE '09)*, Jan. 2009, pp. 1 –2.
- [160] J. Yang, Q. Zhao, and L. Zhang, "The study of frame complexity prediction and rate control in H.264 encoder," in *Proceeding of International Conference on Image Analysis and Signal Processing (IASP 2009)*, Apr. 2009, pp. 187 –191.
- [161] H. Seferoglu and A. Markopoulou, "Distributed rate control for video streaming over wireless networks with intersession network coding," in *17<sup>th</sup> International Packet Video Workshop (PV 2009)*, May 2009, pp. 1 –10.
- [162] S. Puangpronpitag, "Design and performance evaluation of multicast congestion control for the Internet," Ph.D. thesis, University of Leeds, UK, 2003.
- [163] W.-K. Liao and Y.-H. Lai, "Type-aware error control for robust interactive video services over time-varying wireless channels," *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 136 –145, Jan. 2011.
- [164] R. Raghavendra and E. M. Belding, "Characterizing high-bandwidth real-time video traffic in residential broadband networks," in *2010 Proceedings of the 8<sup>th</sup> International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, Jun. 2010, pp. 597 –602.
- [165] M. Pantoja and N. Ling, "A two-level rate control approach for video transcoding," in *Proceedings of the 16<sup>th</sup> IEEE international conference on Image processing*, ser. ICIP'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 3657–3660. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1819298.1819748>

- [166] M. Tun, K. K. Loo, and J. Cosmas, "Rate control algorithm based on quality factor optimization for Dirac video codec," *Image Commun.*, vol. 23, pp. 649–664, Oct. 2008. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1453270.1453642>
- [167] O. A. Lotfallah, M. Reisslein, and S. Panchanathan, "A framework for advanced video traces: evaluating visual quality for video transmission over lossy networks," *EURASIP J. Appl. Signal Process.*, pp. 263–263, Jan. 2006. [Online]. Available: <http://dx.doi.org/10.1155/ASP/2006/42083>
- [168] A. Lie. (2011, Jan.) Evalvid-RA. [Online; accessed 29-Jan-2011]. [Online]. Available: <http://www.item.ntnu.no/arnelie/Evalvid-RA.htm>
- [169] M. Reisslein, L. Karam, and P. Seeling, "H.264/AVC and SVC video trace library: A quick reference guide," National Science Foundation, Tech. Rep., 2009.
- [170] A. Al Tamimi, R. Jain, and C. So-In, "Modeling and prediction of high definition video traffic: A real-world case study," in *Proceeding of 2010 Second International Conferences on Advances in Multimedia (MMEDIA)*, Jun. 2010, pp. 168 –173.
- [171] R. Zhang, R. Ruby, J. Pan, L. Cai, and X. Shen, "A hybrid reservation/contention-based MAC for video streaming over wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 3, pp. 389 –398, Apr. 2010.
- [172] J. Widmer, "Equation-based congestion control for unicast and multicast data streams," Ph.D. thesis, University of Mannheim, 2003.
- [173] —, "Equation-based congestion control," Master thesis, University of Mannheim, 2000.
- [174] I. Bajic and X. Ma, "A testbed and methodology for comparing live video frame rate control methods," *IEEE Signal Processing Letters*, vol. 18, no. 1, pp. 31 –34, Jan. 2011.
- [175] A. S. M. Arif, S. Hassan, O. Ghazali, and S. A. Nor, "The relationship of TFRC congestion control to video rate control optimization," *Proceeding of International Conference on Network Applications, Protocols and Services*, pp. 31–36, 2010.
- [176] X. Lisong and H. Josh, "Media streaming via TFRC: An analytical study of the impact of TFRC on user-perceived media quality," *The International Journal of Computer and Telecommunications Networking*, vol. 51, no. 17, pp. 4744–4764, 2007.

- [177] M. Chen and A. Zakhor, "Multiple TFRC connections based rate control for wireless networks," *IEEE Transactions on Multimedia*, vol. 8, no. 5, pp. 1045–1062, 2006.
- [178] —, "AIO-TFRC: a light-weight rate control scheme for streaming over wireless," in *2005 International Conference on Wireless Networks, Communications and Mobile Computing*, vol. 2, Jun. 2005, pp. 1124 – 1129 vol.2.
- [179] A. Chodorek, "Streaming video with TFRC- simulation approach," in *Joint IST Workshop on Mobile Future, 2004 and the Symposium on Trends in Communications (SympoTIC '04)*, Oct. 2004, pp. 187 – 190.
- [180] O. Heckmann, M. Piringer, J. Schmitt, and R. Steinmetz, "On realistic network topologies for simulation," in *Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, ser. MoMeTools '03. New York, NY, USA: ACM, 2003, pp. 28–32. [Online]. Available: <http://doi.acm.org/10.1145/944773.944779>
- [181] H. Venkataraman, P. Kalyampudi, and G.-M. Muntean, "Cashew: Cluster-based adaptive scheme for multimedia delivery in heterogeneous wireless networks," *Wireless Personal Communications*, pp. 1–20, 2010. [Online]. Available: <http://dx.doi.org/10.1007/s11277-010-0067-8>
- [182] P. Papadimitriou, V. Tsaoussidis, and L. Mamatas, "A receiver-centric rate control scheme for layered video streams in the Internet," *J. Syst. Softw.*, vol. 81, pp. 2396–2412, Dec 2008. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1454787.1454993>
- [183] P. Papadimitriou and V. Tsaoussidis, "SSVP: A congestion control scheme for real-time video streaming," *Computer Networks*, vol. 51, no. 15, pp. 4377 – 4395, 2007. [Online]. Available: <http://www.sciencedirect.com/science/article/B6VRG-4P2YWY9-1/2/a5d0e0acc57a43f22d9a4c221d303ae8>
- [184] —, "A rate control scheme for adaptive video streaming over the Internet," in *Proceeding of IEEE International Conference on Communications ( ICC '07)*, 2007, pp. 616 –621.
- [185] T. de Souza-Daw, N. K. Chilamkurti, and B. Soh, "An integration of H.264 based Error Concealment technique and the SPLIT layer protocol," *Proceeding of International Conference on Mobile Communications and Learning Technologies, Conference on Networking, Conference on Systems*, 2006.
- [186] P. Koshy and S. V. Raghavan, "Quality adaptation for FGS MPEG-4 video streaming over internet," in *Proceedings of the 15<sup>th</sup> international conference on Computer communication*, ser. ICC '02. Washington, DC, USA:

- International Council for Computer Communication, 2002, pp. 654–663.  
[Online]. Available: <http://portal.acm.org/citation.cfm?id=838138.838194>
- [187] S. Boyden, A. Mahanti, and C. Williamson, “Characterizing the behaviour of RealVideo streams,” in *SCS Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, vol. 37, Philadelphia, PA, USA, 2005.
  - [188] J. Byers, M. Handley, G. Horn, M. Luby, and L. Vicisano, “More thoughts on reference simulations for reliable multicast congestion control schemes,” Tech. Rep., 2000. [Online]. Available: <http://www.cs.bu.edu/fac/byers/pubs/mrefsims.ps>
  - [189] M. Anirban, L. E. Derek, and K. V. Mary, “Improving multirate congestion control using a TCP Vegas throughput model,” *Comput. Netw. ISDN Syst.*, vol. 48, no. 2, pp. 113–136, 2005.
  - [190] IEEE Standards Board, “IEEE Recommended Practice for Distributed Interactive Simulation 2013; Verification, Validation, and Accreditation,” *IEEE Std 1278.4-1997 (Reaff 2010)*, pp. 1–78, Jan. 2010.
  - [191] J. Banks, *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. Wiley-Interscience, 1998.
  - [192] S.-P. Chuah, Z. Chen, and Y.-P. Tan, “Channel access allocation for scalable video transmission over contention-based wireless networks,” *IEEE Signal Processing Letters*, vol. 18, no. 1, pp. 15–18, Jan. 2011.
  - [193] F. Bellard, “FFmpeg project,” 2011, [Online; accessed 18-Feb-2011]. [Online]. Available: <http://www.ffmpeg.org/index.html>
  - [194] S. Milani, “Fast H.264/AVC FRExt intra coding using belief propagation,” *IEEE Transactions on Image Processing*, vol. 20, no. 1, pp. 121–131, Jan. 2011.

## Appendix A: Video Samples

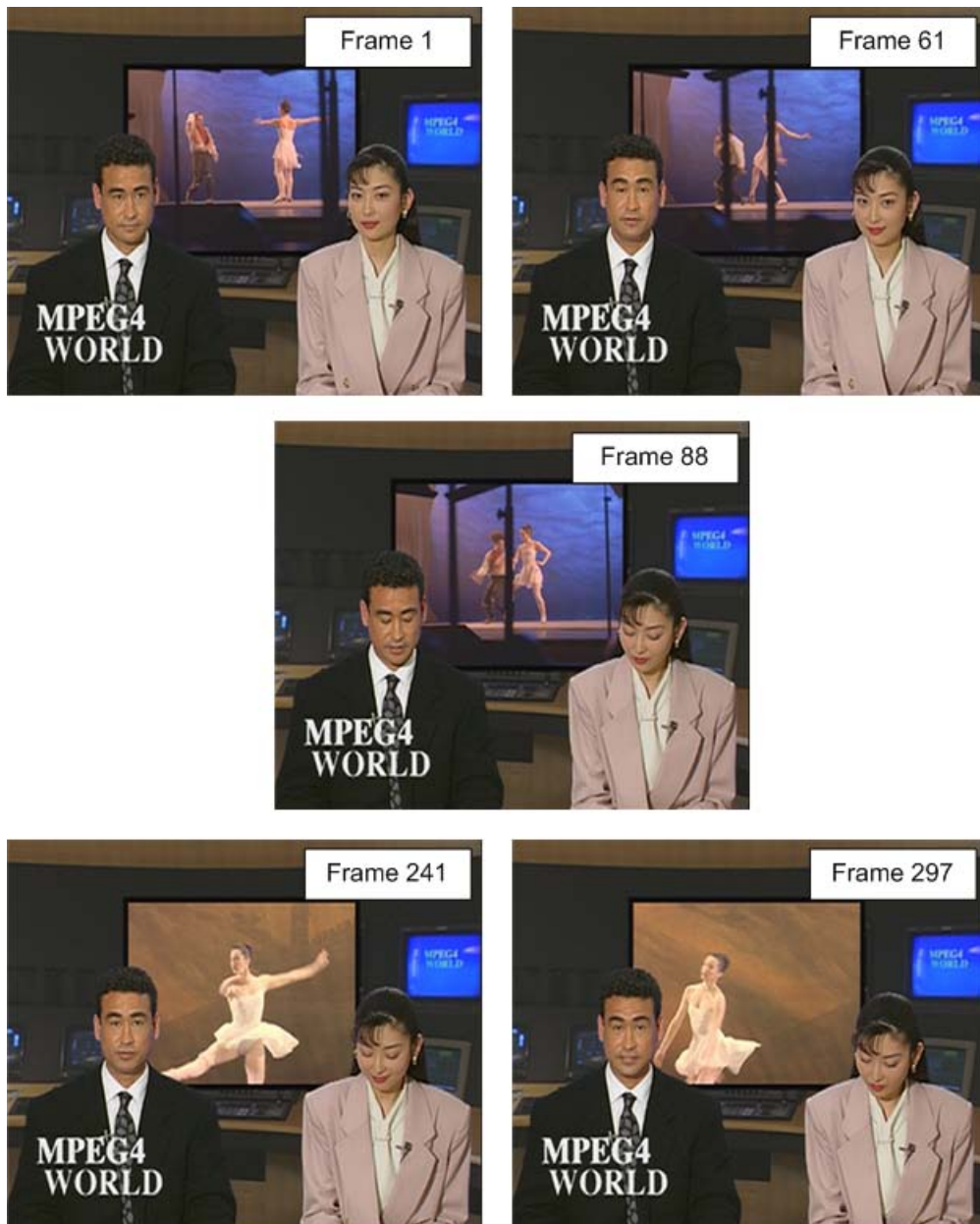


Figure 1: Frames from News sequence

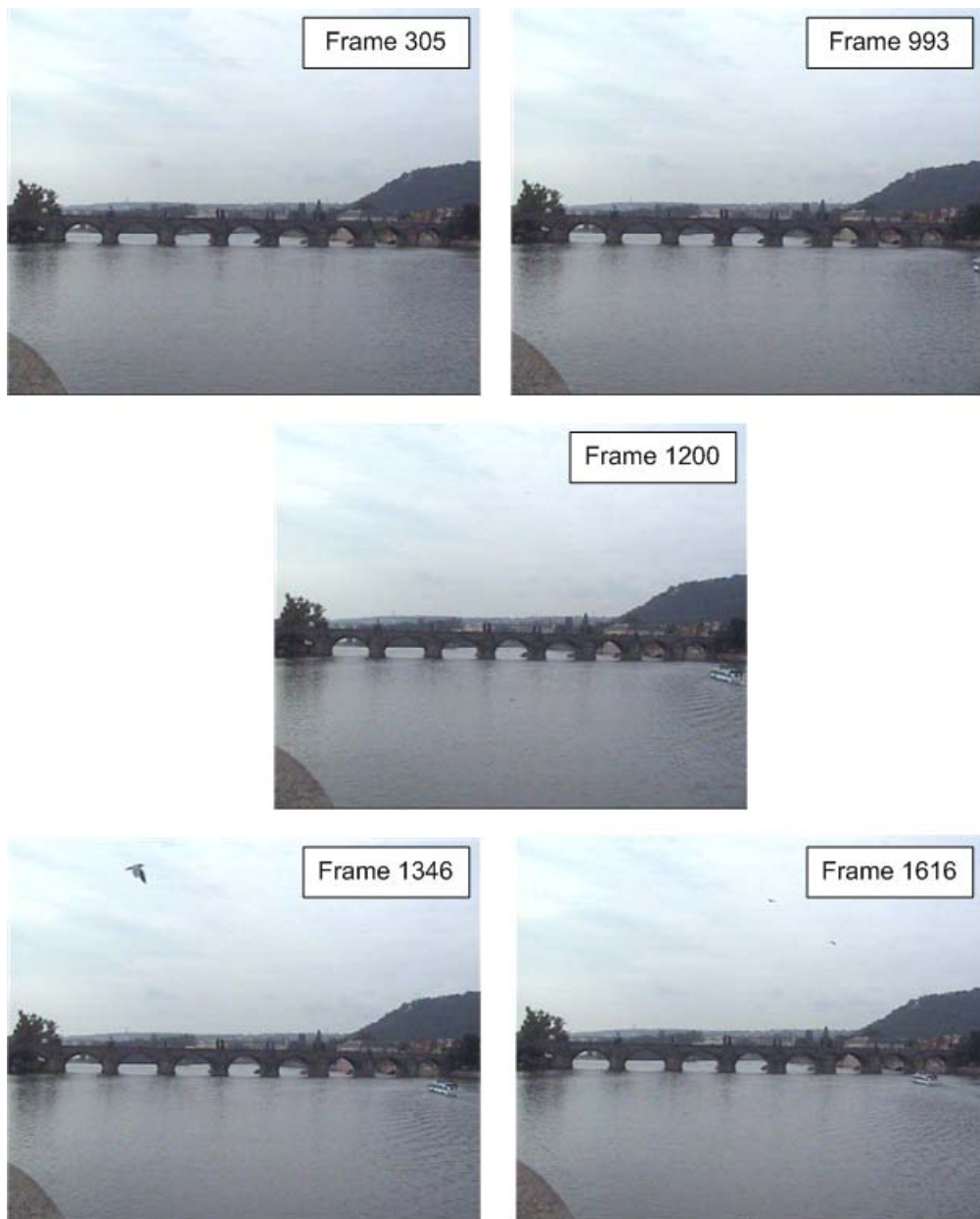


Figure 2: Frames from Bridge (far) sequence

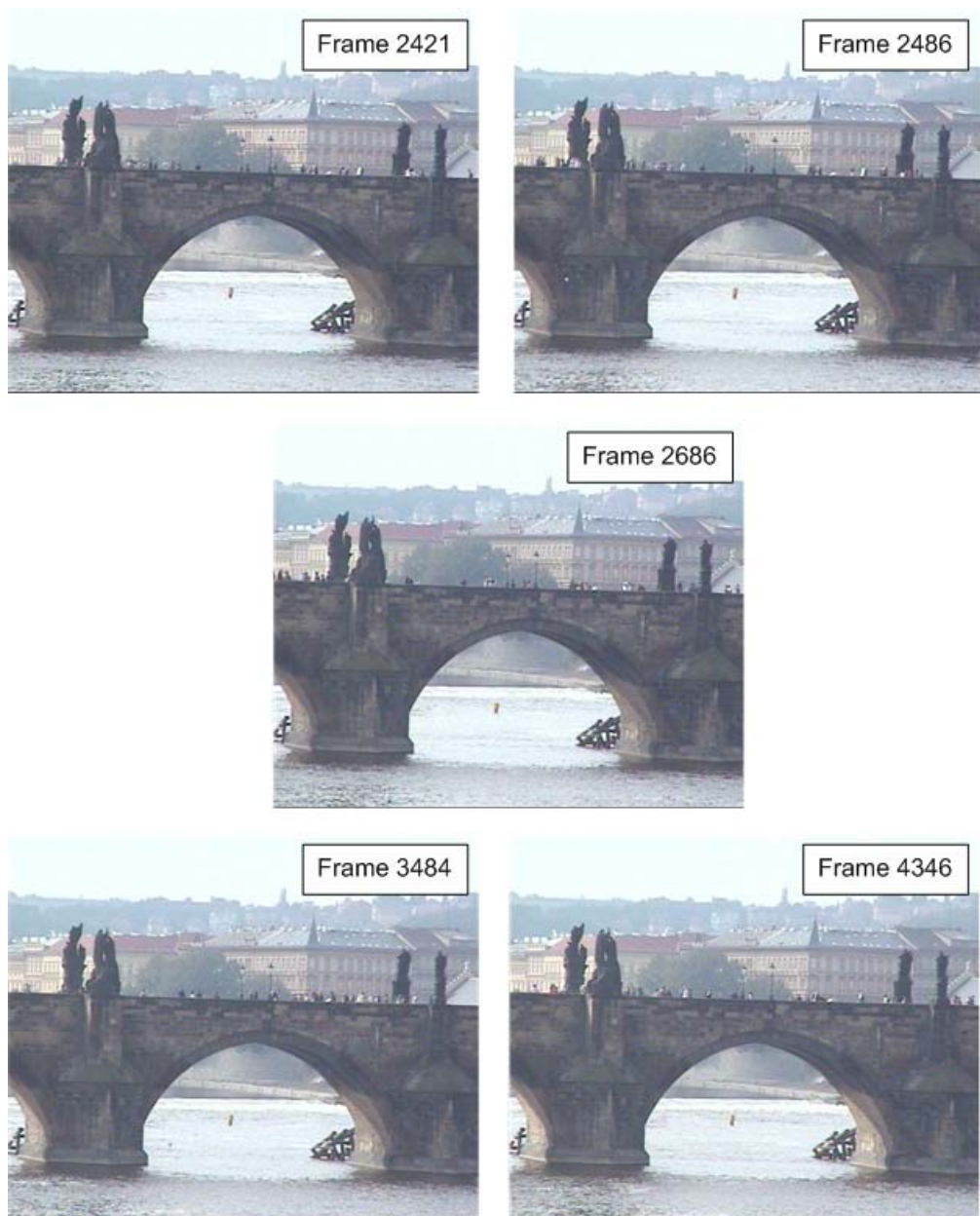


Figure 3: Frames from Bridge (close) sequence





Figure 4: Frames from Bus sequence

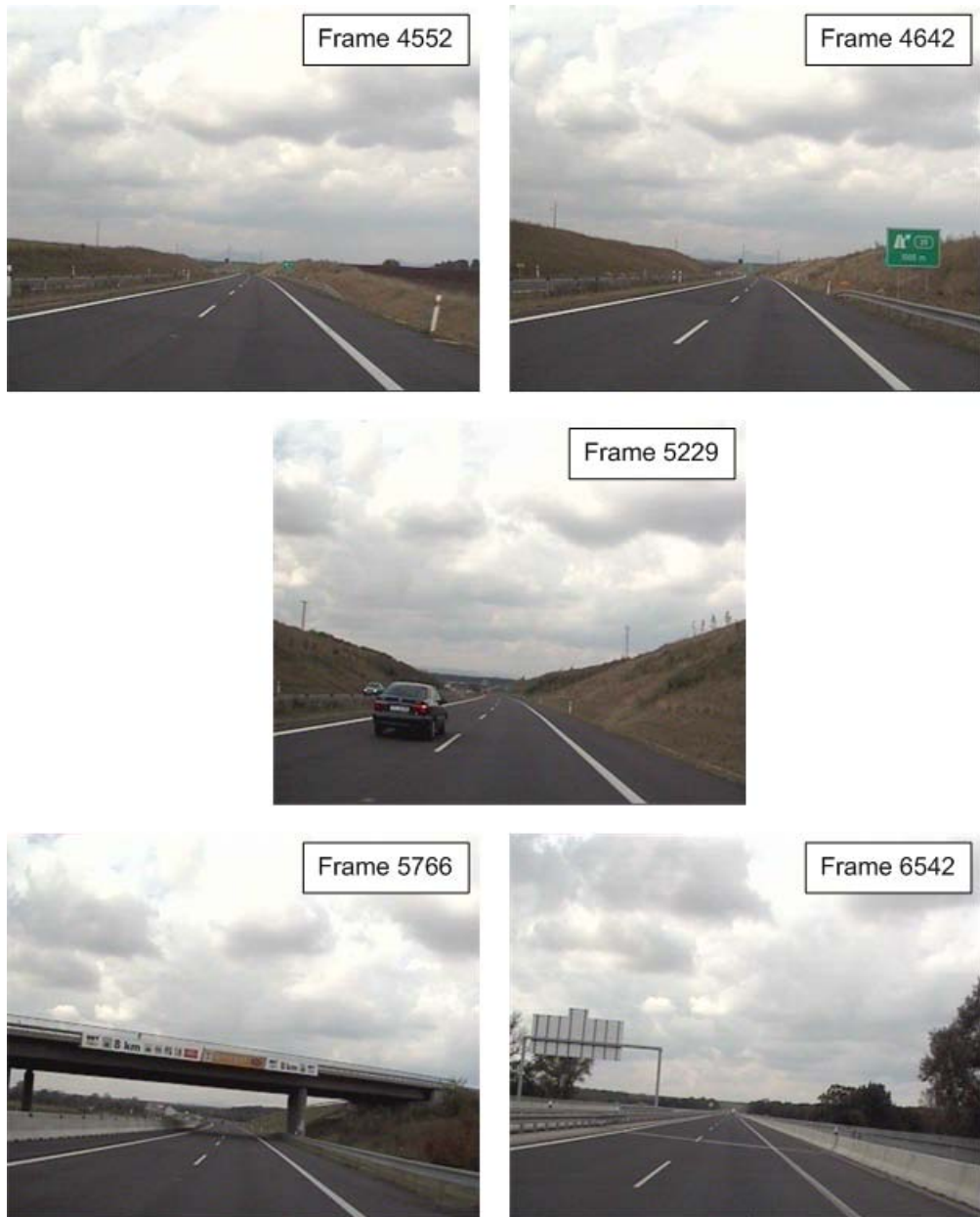


Figure 5: Frames from Highway sequence

## Appendix B: Snapshot of TCL Scripts

```
set ns [new Simulator]

set simtime 404           ;# Simulation time in seconds
set MON_QUEUE_NUM 0
set queueTime 0.4

set cbr_true 0           ;# 0 = SD-SVBR, 1 = CBR
set vbr_rate 0.3333333e6 ;# Oct2010: Average VBR rates for allClips

set max_fragmented_size 972 ;# MTU
set frames_per_second 25    ;
# All video is treated with equal fps in current release

# Link capacities
set access_cap 100Mb        ;# access link capacities
set bottleneck_cap1 1.5e6   ;# between routers
set bottleneck_cap2 1.0e6   ;# between router and destination

# Link propagation delays
set access_del 0.002
set bottleneck_del 0.050

# set number of quantiser scale @ video quality range & packet size
set q_variants 30 ;# number of quantiser scale range
set packetSize [expr $max_fragmented_size + 36]
set mean_psize $packetSize

# This is the size of the queue buffer:
set queueSize1 [expr ($bottleneck_cap1*$queueTime)/(8*$mean_psize)]
set queueSize2 [expr ($bottleneck_cap2*$queueTime)/(8*$mean_psize)]

# SETTING GLOBAL TFRC DEFAULTS:
Agent/TFRC set ss_changes_ 1
Agent/TFRC set slow_increase_ 1
Agent/TFRC set rate_init_ 2
Agent/TFRC set rate_init_option_ 2

Agent/TFRC set SndrType_ 1
Agent/TFRC set oldCode_ false
Agent/TFRC set packetSize_ $packetSize
Agent/TFRC set maxqueue_ 500
Agent/TFRC set printStatus_ true
Agent/TFRC set ecn_ 2 ; # 1 Enable ECN, 2 Disable ECN
Agent/TFRC set useHeaders_ false

#Open the files for nam trace
set f [open out.tr w]
$ns trace-all $f
set nf [open out.nam w]
$ns namtrace-all $nf
```

```

# generate the sending node:
set send_node_1 [$ns node]
set send_node_2 [$ns node]

# generate the routers:
set router_node_1 [$ns node]
$router_node_1 shape "box"
set router_node_2 [$ns node]
$router_node_2 shape "box"

# generate the receiving node:
set recv_node_1 [$ns node]
set recv_node_2 [$ns node]

# define the links between the nodes:
$ns duplex-link $send_node_1 $router_node_1
    $access_cap $access_del DropTail
// we assume server will be at LAN, with 100Mbps speed rate
$ns duplex-link $send_node_2 $router_node_1
    $access_cap $access_del DropTail
$ns simplex-link $router_node_1 $router_node_2
    $bottleneck_cap1 $bottleneck_del DropTail
$ns simplex-link $router_node_2 $router_node_1
    $bottleneck_cap1 $bottleneck_del DropTail
$ns duplex-link $router_node_2 $recv_node_1
    $bottleneck_cap2 $access_del DropTail
$ns duplex-link $router_node_2 $recv_node_
2 $bottleneck_cap2 $access_del DropTail

# set the maximal queue lengths of the routers:
$ns queue-limit $router_node_1 $router_node_2 $queueSize1
$ns queue-limit $router_node_2 $recv_node_1 $queueSize2

#####
# Configure router queue monitor #
#####
set f4 [open outqm.tr w]
set qmon1 [$ns monitor-queue $router_node_1 $router_node_2 $f4 0.02333]
set qmon2 [$ns monitor-queue $router_node_2 $recv_node_1 $f4 0.02333]
[$ns link $router_node_1 $router_node_2] queue-sample-timeout
[$ns link $router_node_2 $recv_node_1] queue-sample-timeout
$qmon1 set pdrops_
$qmon2 set pdrops_

#####
# Set interface between TFRC Receive (of ACKS) #
# and the video rate controller #
#####
Agent/TFRC instproc tfrc_ra {bytes_per_sec backlog} {

$self instvar node_

global vbr1 ns id
set now [$ns now]

```

```

set node_id [$node_id]
#puts "In TFRC instproc. rate = $bytes_per_sec (B/s), node_id = $node_id"

#puts "TCL: before vbr1 TFRC_rateadapt time=[format %9.4f $now]
rate=$bytes_per_sec node=$node_id"
$ns at [expr $now] "$vbr1 TFRC_rateadapt $bytes_per_sec $node_id $backlog"
#puts "TCL: after vbr1 TFRC_rateadapt
#       time=[format %.5f $now] | rate=[format %.2f $bytes_per_sec] |
#       node=$node_id | backlog=$backlog"
}

#####
# Create SD-SVBR traffic sources #
#####
set rng2 [new RNG]
$rng2 seed 0
set xRate [new RandomVariable/Uniform]
$xRate use-rng $rng2
$xRate set min_ 0
$xRate set max_ 2.5e3

# change video trace file name here
puts "SD-SVBR: Start making GOP and Frame trace files
      for $q_variants Rate variants"
for {set i 1} {$i <= $q_variants} {incr i} {
    set original_file_name($i) st_allclips.yuv_Q[expr $i + 1].txt
    set original_file_id($i) [open $original_file_name($i) r]
}

set trace_file_name video2.dat
set trace_file_id [open $trace_file_name w]
set trace_file [new vbrTraceFile_RASV]
$trace_file filename $trace_file_name
set frame_count 0
set last_time 0

# AL: toggle between multiple input files!
# set original_file_id $original_file_id(1)
set source_select 1

set frame_size_file frame_size.dat
set frame_size_file_id [open $frame_size_file w]
for {set i 1} {$i <= $q_variants} {incr i} {
    set frame_size($i) 0
}
set gop_size_file gop_size.dat
set gop_size_file_id [open $gop_size_file w]
for {set i 1} {$i <= $q_variants} {incr i} {
    set gop_size($i) 0
}

set gop_numb 0
set no_ 0
# Convert ASCII sender file on frame size granularity to

```

```

#          ns-2 adapted internal format
while {[eof $original_file_id(1)] == 0} {
    for {set i 1} {$i <= $q_variants} {incr i} {
        gets $original_file_id($i) current_line($i)
    }

    scan $current_line(1) "%d%s%d%s%s%s%d%s"
        no_ frametype_ length_ tmp1_ tmp2_ tmp3_ tmp4_ tmp5_

    set tempStr "%.0f"
    set time [expr 1000 * 1000/$frames_per_second]
    set time [format $tempStr $time]

    if { $frametype_ == "I" } {
        set type_v 1
        set time 0
    }

    if { $frametype_ == "P" } {
        set type_v 2
    }

    if { $frametype_ == "B" } {
        set type_v 3
    }

    # Write to GOP size file after each H-frame found:
    if { $frametype_ == "H" } {
        set puts_string "$gop_numb"
        for {set i 1} {$i <= $q_variants} {incr i} {
            set puts_string "$puts_string $gop_size($i)"
        }
        puts $gop_size_file_id $puts_string
        set gop_numb [expr $gop_numb + 1]
        set type_v 0 ;
        # Must have different type than I-frame
        #          so that the LB(r,b) algorithm finds it!
    }

    flush $gop_size_file_id ;
    #27/12/09: thus it can be read correctly by C++ codes

# Write to frame_size.dat:
    set puts_string "$no_"
    for {set i 1} {$i <= $q_variants} {incr i} {
        set puts_string "$puts_string $gop_size($i)"
    }

    set puts_string "$puts_string $type_v" ;#
    Feb2010: some GoP do not have same size

    puts $frame_size_file_id $puts_string

    for {set i 1} {$i <= $q_variants} {incr i} {
        scan $current_line($i) "%d%s%d%s%s%s%d%s"
            no_ frametype_ length_($i) tmp1_ tmp2_ tmp3_ tmp4_ tmp5_

```

```

        #puts "no_ frametype_ length($i)
        #      tmp1_ tmp2_ tmp3_ tmp4_ tmp5_ =
        #      $no_ $frametype_ $length($i)
        #      $tmp1_ $tmp2_ $tmp3_ $tmp4_ $tmp5_"
        set gop_size($i) [expr $gop_size($i) + $length($i) ]
    }

# Write to video2.dat:
    set puts_string "$time $length_ $type_v $max_fragmented_size"
    for {set i 2} {$i <= $q_variants} {incr i} {
        set puts_string "$puts_string $length($i)"
    }
    puts $trace_file_id $puts_string
    incr frame_count
    # puts "Frame count = $frame_count,
    #      trace id=$trace_file_id dan put string=$puts_string"
}

flush $frame_size_file_id
puts "SD-SVBR: #of frames written to GOP and
      Frame trace files: $frame_count"
close $trace_file_id ;
# Note that this new trace file is closed for writing and
# opened below for reading through being a new Tracefile
#      in eraTraceFile2::setup()

#####
# Add SD-SVBR sources                                     #
#####

set tfrc0 [new Agent/TFRC]
$ns attach-agent $send_node_1 $tfrc0
set vbr1 [new Application/Traffic/eraVbrTrace_RASV]
$vbr1 attach-agent $tfrc0
    # puts "Started adding SVBR source"
$tfrc0 set packetSize_ $packetSize ;
    # this is the MSS for the TFRC
$tfrc0 set TOS_field_ 1 ;
    # New 120905: tag ECF enabled sources! 0 is default.
$tfrc0 set_filename sd_be_0 ;
    # Connect a file name to TFRC source to write transmit trace da
$tfrc0 set ecn_ 2 # May2010: disable ECN

$vbr1 set running_ 0
set offRate [$xRate value]

    $vbr1 set r_ [expr $vbr_rate]
    $vbr1 attach-tracefile $trace_file
    $vbr1 set b_ 1.0e6
    $vbr1 set q_ 10
    $vbr1 set GoP_ 12
    $vbr1 set fps_ $frames_per_second

$tfrc0 set class_ 4

```

```

    $tfrc0 set fid_ 1
    $ns color 1 blue
    #puts "in TCL: b4 sink"
    set sink0 [new Agent/TFRCSink]
    #puts "in TCL: after sink"
    $ns attach-agent $recv_node_1 $sink0
    $ns connect $tfrc0 $sink0
    $sink0 set_trace_filename rd_be_0 ;
        # Connect a file name to TFRC sink to

    puts "Completed adding SVBR source"

#####
## Simulation Main
#####
set rng3 [new RNG]
set vbrStart [new RandomVariable/Uniform]
$vbrStart use-rng $rng3
$vbrStart set min_ 0.040
$vbrStart set max_ 4.000

    set startTime [$vbrStart value]

    $send_node_1 label "Srv 1"
    $send_node_2 label "Srv 2"
    $router_node_1 label "R1"
    $router_node_2 label "R2"
    $recv_node_1 label "Client 1"
    $recv_node_2 label "Client 2"

    $ns at 0.010 "$vbr1 start"
    $ns at $simtime "$vbr1 stop"

proc finish {} {
    global ns f nf f0 f1 f2 f3 f4 f5
        qfile tchan2_ qmon1 qmon2 bottleneck_cap1 bottleneck_cap2
        simtime outfile queueSize1 queueSize2

    # $ns flush-all
    $qmon1 instvar bdrops_ bdepartures_
    set utlzn [expr ($bdepartures_ * 8.0)/
        ($bottleneck_cap1 * $simtime)]
    set d [expr 1.0*$bdrops_ / ($bdrops_ + $bdepartures_)]
    puts "\n##### statistics #####"
    puts "#Bytes of drops R1      : $bdrops_ "
    puts "#Bytes of departures R1 : $bdepartures_ "
    puts "drops stats              : $d "
    puts "utilization                 : $utlzn "

    $qmon2 instvar bdrops_ bdepartures_
    set utlzn [expr ($bdepartures_ * 8.0)/($bottleneck_cap2 * $simtime)]
    set d [expr 1.0*$bdrops_ / ($bdrops_ + $bdepartures_)]
    puts "\n##### statistics R2 #####"

```



```

    puts "#Bytes of drops R2      : $bdrops_ "
    puts "#Bytes of departures R2 : $bdepartures_ "
    puts "drops stats              : $d "
    puts "utilization              : $utlzn "
    puts "                        "

    $ns flush-trace
    close $f
    close $f4
    close $nf
    exit 0
}

$ns at [expr $simtime + 1.0] "finish"

$ns run

```

## Appendix C: Snapshot of C++ Codes

```
static class vbr_rateadapt_RASVClass : public TclClass {
public:
    vbr_rateadapt_RASVClass() :
        TclClass("Application/Traffic/eraVbrTrace_RASV") {}
    TclObject* create(int, const char*const*) {
        return(new vbr_rateadapt_RASV());
    }
} class_vbr_traffictrace;

void vbr_rateadapt_RASV::start()
{
    init();
    running_ = running2_ = 1;
    nextPkttime_ = next_interval(size_);
    timer_.resched(nextPkttime_);
}

void vbr_rateadapt_RASV::init()
{
    int i, pre_q;
    char bacaBaris[1000], *bacaData;
    int habis_data=0, jumGops=548;
    // 27/1/2010 - anggapkan maksimum Gop=547
    FILE *f_gop, *f_frame;
    totalPackets = totalFrames = 0;
    //char* f_arr_c[10000];

    vbrFile = fopen("out.vbrRate_RASV","w");
    // Feb2010: just for tracing
    gopFile = fopen("out.gopData_RASV", "w");
    // April2010: to record the GoP size
    // tujuan untuk dapat next GoP size =
    // q*R_open tetapi base kepada Gop_size.dat
    if((f_gop = fopen("gop_size.dat", "r")) == NULL) {
        printf("can't open file %s\n", f_gop);
    }

    fgets(bacaBaris,1000,f_gop);
    //skip baris pertama - dummy data
    int l=1; //mula data sebenar dgn baris 2
    while (fgets(bacaBaris,1000,f_gop)!=NULL) {
        bacaData = strtok(bacaBaris, " ");
        /* read the first data in the line */
        q_arr[l][1] = atoi(bacaData);
        for (int j=2; j<=31; j++) {
            bacaData = strtok(NULL, " ");
            /* read the rest of the data */
            q_arr[l][j] = atoi(bacaData);
        }
    }
}
```

```

        l++;
    }
    fclose (f_gop);

    if((f_frame = fopen("frame_size.dat", "r")) == NULL) {
        printf("can't open file %s\n", f_frame);
    }

    fgets(bacaBaris,1000,f_frame);
        //skip baris pertama – dummy data
    l=1; //mula data sebenar dgn baris 2
    while (fgets(bacaBaris,1000,f_frame)!=NULL) {
        bacaData = strtok(bacaBaris, " ");
        /* read the first data in the line */
        f_arr[l][1] = atoi(bacaData);
        for (int j=2; j<=31; j++) {
            bacaData = strtok(NULL, " ");
            /* read the rest of the data */
            f_arr[l][j] = atoi(bacaData);
        }
        bacaData = strtok(NULL, " ");
        /* Feb2010: read the frametype */
        f_arr[l][32] = atoi(bacaData);
        l++;
    }
    fclose (f_frame);

    int saiz_frame=972;
    int bil_packets=0;
    int gop_indeks , n_;
    for (int m=2; m<=31; m++) {
        n_=1;
        gop_indeks=1;
        if ((f_arr[n_][m])<saiz_frame)
            bil_packets=1;
        else
            bil_packets=ceil((f_arr[n_][m])/
                (double) saiz_frame);

        for (n_=2; n_<=l; n_++) {
            if ((f_arr[n_][m]-f_arr[n_-1][m])<saiz_frame)
                bil_packets+=1;
            else
                bil_packets+=ceil((f_arr[n_][m]-
                    f_arr[n_-1][m])/(double) saiz_frame);

            if (f_arr[n_][32]==0) {
                saiz_gop_w_oh[gop_indeks][m]=
                    (bil_packets*1008)-1008;
                gop_indeks++;
                bil_packets=0;
            }
        }
    }

```

```

    r_ = r_ * GoP_ / (8*fps_) ;
    // r_ now has dimensions of bytes/GOP
    b_ = b_ * GoP_ / (8*fps_); // b_ in bytes
    lb_X = b_ * 0.5; // Init buffer to half value
    lb_R = 0.0;

    /* 28/12/09 */
    lb_R_wanted = saiz_gop_w_oh[1][(int)q_];
    //Nov2010: as suggested by Hamdi
    printf("lb_X=%f8.4, lb_R_wanted = %f8.4\n",
        lb_X, lb_R_wanted);
    int jumpa=0;
    int pusingan=2;
    while (pusingan<=31 && jumpa==0) {
        saiz_gop[pusingan] = saiz_gop_w_oh[1][pusingan];
        if (saiz_gop[pusingan]<lb_R_wanted)
            jumpa=1;
        else
            pusingan++;
    }

    if (pusingan==2)
        pre_q=2;
    else if (pusingan>31)
        pre_q=31;
    else {
        if ((saiz_gop[pusingan-1]-lb_R_wanted)<
            (lb_R_wanted-saiz_gop[pusingan])) {
            if (saiz_gop[pusingan-1]==saiz_gop[pusingan-2])
                pre_q=pusingan-2;
            else
                pre_q=pusingan-1;
        }
        else
            pre_q=pusingan;
    }
    Q = pre_q; // July2010: using calculated q_
    // that satisfied LB algorithm for stored media
    if (Q > 31) //no more this issue here
        Q = 31;
    Q_flag = 0;
    numbFrames = 0;
    frameIndex = 0;
    numbGops = 0;
}

void vbr_rateadapt_RASV::timeout()
{
    unsigned long x_, y_, i;
    int size_w_oh; // Size with overhead
    double e1, e2, e1_x, e2_x;

```

```

    double Rtemp, Qtemp, r_open;
    double x_temp, pre_lb_X;
    int bilPacket_y=0;
int l, habis_data=0, jumGops=548;
    //25/12/09 - anggapkan maksimum Gop=547
    int lastGop = 547; //Feb2010 : if not the last
        // will get the GoP after the last, which is none.
int pre_q; //July2010: don't want to change the initial q_
    int bucketLow=0; //August2010: to break Hamdi limitation

    if (! running_)
        return;

    x_=size_/max_; // number of complete (full sized) IP packets
    y_=size_%max_; // the remaining rest part
    if (y_>0) bilPacket_y=1;
    //Feb2010: for easy tracing
    // (bottleneck/inefficiency in the network)
totalPackets = totalPackets+x_+bilPacket_y;

    if(f_==0 || f_==1){
        agent_ ->set_prio(10);
    }
    else if(f_==2){
        agent_ ->set_prio(11);
    }
    else if(f_==3){
        agent_ ->set_prio(12);
    }
    else {
        agent_ ->set_prio(f_);
    }

    if (f_ != 0) {
        numbFrames++;
        totalFrames++; //Feb2010: to trace performance...
    }
#endif TFRC
    agent_ ->set_pkttype(PT_VIDEO);
#endif
    agent_ ->set_frametype(f_);

    // Here is the Tx of packets ,
    // all packets of same frame is sent at *same* time
    // First the max size packets (if any)
    size_w_oh = 0;
    if(x_ > 0){
        for(i=0 ; i<x_ ; i++) {
            agent_ ->set_quant(Q);
            agent_ ->sendmsg(max_+HDR_SIZE);
            size_w_oh += max_+HDR_SIZE;
        }
    }
    // here goes the rest packet (normally, as in H264,

```

```

//      the packets are divided on slice boundary)
if (y_!=0) {
    agent_ ->set_quant(Q);
    agent_ ->sendmsg(y_+HDR_SIZE);
}

if (f_==0) {
    numbGops++;
}

double masa = Scheduler::instance().clock();
// ASMATT Feb2010; just to examine current time
printf("Before calc: masa=%9.4f, Q=%d, f_=%d, numbGops=%d,
lb_R=%4.3f, size_w_oh=%d, numbFrames=%d,
totalFrames=%d, size_:%4.3f, PktNo:%d\n",
masa, Q, f_, numbGops, lb_R, size_w_oh,
numbFrames, totalFrames, (double)size_,
totalPackets);
//Feb2010: print to file for performance/efficiency tracing
fprintf(vbrFile, "%9.5f %5d %7d %4d %4d %4d %4d\n",
masa, totalPackets, size_w_oh, numbFrames,
numbGops, totalFrames, Q);

//July2010: print data before calculate the algorithm
pre_lb_X = lb_X;
//July2010: keep the old X @ X(k-1) for tracking purpose
masa = Scheduler::instance().clock();
//Feb2010; just to examine current time
printf("\nBefore eq 5: Time=%9.4f, numbGops=%d,
pre_lb_X=%4.3f, r_=%4.3f, b_=%4.3f, lb_R=%4.3f,
Q=%d, q_=%d\n", masa, numbGops, pre_lb_X, r_, b_,
lb_R, Q, (int)q_);

// July2010: Calculate X(k) @ lb_X
lb_X -= r_; // Eq. 5
lb_R -= 1008;
//April2010 - remove the last packet which
// is actually the header packer for the next GoP
if (lb_X < 0.0) {
    lb_X = 0.0; // Eq. 5
}
lb_X += lb_R; // Eq. 5
if (lb_X > b_) {
    lb_X = b_; // Eq. 5
}

// July2010: Calculate e
x_temp = lb_X/b_;
e1 = e2 = x_temp;
//July2010: buffer fullness function, either near to b_ or 0
if (x_temp > 1.0) x_temp = 1.0;

//July2010: Calculate R_open(k)
r_open = saiz_gop_w_oh[numbGops][(int)q_];

```

```

//Nov2010: Calculating R(k)
float paras = 0.75;
int active = 0; //Nov2010
if (r_open < (r_+((b_/2 - r_)/2))) {
    paras=0.75;
    //Dis2010 - Higher than this lead not less stable QP
    active=1;
}
else {
    if (r_open < b_/2) {
        paras=0.75; //previous 0.6667;
        active=2;
    }
    else {
        paras=0.5;
        //Dis2010; for a highly high data,
        // this might be not low enough,
        // might be beneficial if create
        // another active categories > b_
        active=3;
    }
}

if (lb_X > r_) {
    Rtemp = (paras*b_-(lb_X-r_));
    //Nov2010: modified eq. 8 to limit
    // the boundaries into b
}
else
    Rtemp = paras*b_; //Nov2010

if (Rtemp<r_) Rtemp=r_;

//July2010: Determine Q(k)
lb_R_wanted = Rtemp;
printf("lb_R_wanted = %8.4f\n", lb_R_wanted);

int jumpa=0;
int pusingan=2;
while (pusingan<=31 && jumpa==0) {
    saiz_gop[pusingan] =
        saiz_gop_w_oh[numbGops][pusingan];
    if (saiz_gop[pusingan]<lb_R_wanted)
        jumpa=1;
    else
        pusingan++;
}

if (pusingan==2)
    pre_q=2;
else {
if (((saiz_gop[pusingan-1]-lb_R_wanted)<
    (lb_R_wanted-saiz_gop[pusingan])) || pusingan>31) {
    if (pusingan>31) pre_q=31;
}
}

```

```

        int loop = 1;
        pre_q=pusingan-1;
        int undur = 1;
        while (loop==1) {
            printf("Dlm loop: undur=%d, pusingan=%d,
                    saiz1=%d, saiz2=%d\n", undur, pusingan,
                    saiz_gop[pusingan-undur],
                    saiz_gop[pusingan-(undur+1)]);
            if (saiz_gop[pusingan-undur]!=
                saiz_gop[pusingan-(undur+1)]) {
//Nov2010: sometimes previous GoP
//      same size with the current GoP
            pre_q=pusingan-undur;
                loop=0;
                break;
            }
            undur++;
        }
        loop=0;
    }
    else
        pre_q=pusingan;
}

//Dec2010: Smoothing codes
int old_Q = Q;
int saiz_gop_Q = saiz_gop_w_oh[numbGops][Q];
int next_X = (saiz_gop_Q-r_)+lb_X;
int next_Xpreq = (saiz_gop[pre_q]-r_)+lb_X;
printf("before smoothing Q; Q=%d, next_X=%d, next_Xpreq=%d,
        lb_X=%d\n", Q, next_X, next_Xpreq, (int)lb_X);
if (pre_q==Q+1 || pre_q==Q+2) {
//Nov2010: to stabilize QP value
if ((active==2) || (active==3))
    if (next_X <= (0.83333*b_)) {
        pre_q = (int)Q;
    }
}

if (pre_q==Q-1 || pre_q==Q-2) {
    if ((active==2) || (active==3))
        if (next_X >= (b_/2))
            pre_q = Q;
}

if (Q-pre_q < 5 && Q-pre_q > 0) {
    if ((active==1) && (next_X >= (b_/2)) &&
        (next_X!=next_Xpreq))
//Dec2010: there will be some data previous=current
        pre_q = Q;
}

if (pre_q-Q < 5 && pre_q-Q > 0) {
    if ((active==1) && (next_X <= (0.83333*b_)))

```



```

        pre_q = Q;
    }

    if (Q-pre_q >= 5) {
        if ((active==1) && (next_Xpreq > (0.6667*b_))) {
            if ((saiz_gop[pre_q]-saiz_gop_Q) > (0.08333*b_)) {
                pre_q = ((Q-pre_q)/2)+1+pre_q;
                printf("test smoothing pre_q=%d\n", pre_q);
            }
        }
    }
    //smoothing codes tamat

    int Q2 = pre_q;

    double waktu = Scheduler::instance().clock();
    //Feb2010; just to examine current time
    printf("After calc: waktu=%4.3f, lb_X=%4.3f,
        lb_R=%4.3f, e1=%4.3f, R_open=%4.3f,
        Rtemp=%4.3f, pre_q=%d\n",
        waktu, lb_X, lb_R, e1, r_open,
        Rtemp, pre_q);

    // simpan to array
    QnSize_arr[numbGops-1][1] = pre_q;
    QnSize_arr[numbGops-1][2] = saiz_gop_w_oh[numbGops-1][pre_q];

    Q = pre_q;
    if (Q < 2) {
        Q = 2;
    }
    if (Q > 31) {
        Q = 31;
    }

    numbFrames = 0;
}

/* figure out when to send the next one */
nextPkttime_ = next_interval(size_);
/* fetch both next interval and size of next frame */
/* now, the nextPkttime_ contain how long time
    to start next frame transmission */
/* schedule it */
timer_.resched(nextPkttime_);
}

```