A STUDY OF RATE CONTROL FOR H.265/HEVC VIDEO COMPRESSION

by

Reese Eric Childers

A thesis presented to the Department of Computer Science
and the Graduate School of the University of Central Arkansas
in partial fulfillment of the requirements for the degree of

Master of Science
In
Applied Computing

Conway, Arkansas
August 2020

TO THE OFFICE OF GRADUATE STUDIES:

The members of the Committee approve the thesis of

Reese Eric Childers presented on

07/09/2020.

Yu Sun

Digitally signed by Yu Sun
DN: cn=Yu Sun, o=University of Central Arkansas, ou=Computer Science Dept., email=yusun@uca.edu, c=US
Date: 2020.07.09 15:17:24 -05'00'

Committee Chairperson

Sinan Kockara

Digitally signed by Sinan Kockara
Date: 2020.07.10 01:03:53 -05'00'

Committee Member

Vamsi Paruchuri

Digitally signed by Vamsi Paruchuri
DN: OU=CS, O=UCA, CN=Vamsi Paruchuri, E=vparuchuri@uca.edu
Reason: I am approving this document with my legally binding signature
Location: your signing location here
Date: 2020-07-12 11:33:04
Foxit Reader Version: 9.7.0

Committee Member

PERMISSION

Title A STUDY OF RATE CONTROL FOR H.265/HEVC VIDEO COMPRESSION

Department Computer Science

Degree Master of Science

In presenting this thesis in partial fulfillment of the requirements for a graduate degree from the University of Central Arkansas, I agree that the Library of this University shall make it freely available for inspections. I further agree that permission for extensive copying for scholarly purposes may be granted by the professor who supervised my thesis work, or, in the professor's absence, by the Chair of the Department or the Dean of the Graduate School. It is understood that due recognition shall be given to me and to the University of Central Arkansas in any scholarly use which may be made of any material in my thesis.

Reese Eric Childers Digitally signed by Reese Eric Childers
Date: 2020.07.16 18:32:48 -05'00'

07/16/2020

Date

ACKNOWLEDGEMENT

I would like to thank my advisor Dr. Sun for her encouragement and support over the course of my academic career. I deeply appreciate everything you have done for me, the knowledge that you have shared, and the opportunities you have provided.

I would also like to thank Dr. Kockara and Dr. Paruchuri for their time, advice, review and suggestions of this thesis.

ABSTRACT

The objective of Rate Control (RC) is to achieve optimum visual quality for video applications under some real-world constraints, e.g. bandwidths and delay. Aiming at improving rate control performance, our study concentrated on current rate control techniques for H.265/HEVC - the current international video compression standard. In this thesis, we present a new bufferless direct rate regulation algorithm for H.265/HEVC video compression. To achieve accurate bitrate control for real-time networked video applications based on the Proportional-Integral-Derivative (PID) control theory, our algorithm first performs target bit allocation, and then directly controls the number of actual compression bits to get close to the number of target encoding bits. Different from conventional rate control approaches, a buffer is not adopted in our algorithm which naturally reduces encoding delay and improves real-time response. When compared with the rate control scheme adopted by H.265/HEVC reference software, the experimental results have demonstrated that, on average, our proposed algorithm obtains higher encoding quality up to 3.74 dB, while improving rate control accuracy up to 23.71%.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS

AVC	Advanced Video Coding
CBR	Constant Bit Rate
CSV	Comma-Separated Values
CTU	Coding Tree Unit
CU	Coding Unit
DCT	Discrete Cosine Transformation
GOP	Group of Pictures
HEVC	High Efficiency Video Coding
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
ITU	International Telecommunications Union
JCT-VC	Joint Collaborative Team on Video Coding
JVT	Joint Video Team
MAD	Mean Absolute Difference
MB	Macroblock
MC	Motion Compensation
ME	Motion Estimation
MPEG	Motion Picture Experts Group
MV	Motion Vector
OBT	Optimal Bit Allocation
PID	Proportional-Integral-Derivative
PSNR	Peak Signal-to-Noise Ratio
QP	Quantization Parameter
RC	Rate Control
R-D	Rate-Distortion
RDO	Rate-Distortion Optimization
R-Q	Rate Quantization
RTE	Recursive Taylor Expansion
VBR	Variable Bitrate
VCD	Video CD
VCEG	Video Coding Experts Group

CHAPTER 1 VIDEO COMPRESSION BASICS

Video streaming has become a prolific source of media on the Internet with over fifty percent of all downstream internet traffic being video streaming data. With this popularity, a rapid rise in video quality going beyond HD and into 4k and 8k resolutions have been emerging. Simultaneously, video streaming services such as Netflix and YouTube have become massively popular which contributes to the overwhelming percentage of video related data being streamed. As more video streaming services enter the market, the popularity of streaming high-quality video will only increase and demand for better compression methodologies along with it. The need for video compression has been prevalent even in the earliest days of the internet for services such as video conferencing and phones or for fitting media on storage devices like CD-ROMs. A broad range of applications with various constraints, e.g. bandwidth and delay, must be taken into account when compressing video. To ensure these constraints are met, a rate control (RC) system must be employed in video compression to obtain optimum visual quality for video applications under some real-world constraints. As the demand and quality of video streaming services and beyond increase, video compression and the underlying technologies must evolve with it to enable these services to operate effectively.

The emergence of high resolution video, in particular 4k and 8k, was the primary driving force behind the introduction of the High Efficiency Video Coding (H.265/HEVC) standard with the goal being a 50% performance gain over its predecessor H.264/AVC at the same video quality level [1]. While H.264/AVC has been the backbone for numerous video streaming platforms over many years, a new standard was highly demanded that could offer a higher rate of compression for these new resolution videos [1]. Due to its goal of a 50% performance gain, H.265/HEVC has introduced several new technologies to meet this objective outlined later in this paper. Though H.265/HEVC stipulates the compression techniques and decoder design, it does not provide specific methods for some components such as rate control. Because there is no set standard, various methodologies and implementations can be proposed and incorporated into future versions of HEVC.

As the need for video compression grows and the standards for video compression progress, new methodologies for handling specific aspects of video compression, such as rate control, have been progressing as well. In this chapter, we review the principles of video compression, discuss international compression standards, and introduce the basic concept of video rate control - a crucial component of video compression.

1.1 Basic Principles of Video Compression

Since the size of video data is huge, video compression is a necessity for various video applications. Without compression, even a low resolution video can take up a large amount of data. Let's take the following video for example: the resolution (frame size) is 640x480 pixels, 24 bits/pixel, the frame rate is 30 frames per second (fps), and the video length is 5 minutes. The transmission rate or the bandwidth of such a low resolution 5-minute video would be around 27 MB/s and would take a total space of 8.1 GB. Higher resolution videos such as 4k take substantially higher amounts of space to store. A video that has a resolution of 4096x2160 with 30 fps and a running time of 10 minutes would take 477 TB to store and 796 GB/s (bandwidth) to transmit. Due to video requiring such an enormous amount of space and bandwidth, as well as the need to transmit videos quickly, video compression must be employed to significantly reduce the huge amount of space/bandwidth required when storing or transmitting videos.

There is a high degree of redundancy in video data which can be expressed in two forms: spatial redundancy and temporal redundancy. Many pixels in a frame are oftentimes repeated or very similar to other adjacent pixels. This is known as spatial redundancy. It refers to the correlation among neighboring pixels within the same frame. Consecutive frames in a video are also usually similar and temporal redundancy refers to the correlation between successive frames. Video compression, as discussed below, can be achieved through reducing spatial redundancy and temporal redundancy [2].

1.1.1 Temporal Redundancy Removal - Motion Estimation & Compensation

As mentioned previously, temporal redundancy refers to the similarity between successive frames. Within a video sequence, individual frames tend to change very little aside from what motion takes place. Thus, the basic idea of removing temporal redundancy is to take the difference between two consecutive frames and only encode the difference to save bits. From Figure 1.1, we can see that the values in a difference frame (original frame-previous frame) are much smaller than the pixel values in an original frame. It is well known that a smaller value needs fewer binary bits to represent when compared with a bigger value. Thus, we can save a lot of binary bits in representing the difference. For example, only 2 binary bits are needed to represent the difference value *3* (1st row & 1st column) in Figure 1.1 (b), while 8 binary bits must be used to represent its corresponding original value *139* (1st row & 1st column) in Figure 1.1 (a). Compression algorithms can choose an anchor frame to use as a reference frame, find the difference for subsequent frames, and then code only the differences to save bits. However, if we subtract two consecutive frames directly and only encode the difference, the compression ratio is around 3:1, which is very low.

139	137	127	133	132	136	134	136
138	140	121	133	128	129	139	140
130	124	125	136	129	128	131	121
132	132	131	127	129	128	123	121
186	136	124	140	125	136	129	135
136	137	124	132	124	120	127	135
138	137	121	129	126	136	123	123
185	139	135	136	125	130	140	122

(a) Original Frame

3	9	0	0	0	0	0	0
9	-8	0	-9	0	8	8	3
-3	0	-6	0	-5	0	-3	-7
0	0	-4	0	2	0	2	-1
0	3	0	-7	0	-4	2	0
9	0	4	-6	6	0	0	0
10	0	0	0	0	-9	0	4
0	-6	0	0	0	-3	-9	0

(b) Difference Frame

Figure 1.1: Original Frame and the Difference Frame

In order to obtain higher compression performance, Motion Estimation (ME) and Compensation have been developed to minimize the difference and reduce temporal redundancy to the highest degree. Motion estimation finds the minimum difference in the two frames through a spatial search that is performed for each macroblock of a frame [3]. That is to say, ME searches for the best match, with the minimum difference, for macroblocks between the current frame and the reference frame. The direction and type of transformation is saved as a motion vector (MV) and any difference between the translated macroblocks is accounted for in a residual frame.

1.1.2 Spatial Redundancy Removal - Discrete Cosine Transformation (DCT)

Discrete Cosine Transformation (DCT) converts the information contained in 8x8 blocks of pixels from the spatial domain to the frequency domain. After DCT, we lose the spatial information (pixel values), but we obtain the frequency information (frequency coefficients) instead. As we can see from Figure 1.2 (b), the DC coefficient (1st row & 1st column) and the first few low frequency coefficients in the upper-left corner have larger absolute values than the remaining high frequency coefficients. This indicates that DCT compacts the most energy of a frame into the first few coefficients (upper-left part) as possible. That is also to say, that most information of a video frame is accurately described by the first few low frequency coefficients. Since humans are much less likely to notice the loss of higher frequency components than the loss of lower frequency components, the spatial redundancy can be removed by encoding the lower frequencies with more bits than the higher frequencies. If most information is accurately described by the first few coefficients we can use more bits to finely encode the first several low frequency coefficients while using fewer bits to coarsely compress the remaining high frequency coefficients, or even simply discard them without encoding. By doing this, a lot of bits are saved with little signal distortion, and thus the spatial redundancy can be removed.

139	137	127	133	132	136	134	136
138	140	121	133	128	129	139	140
130	124	125	136	129	128	131	121
132	132	131	127	129	128	123	121
186	136	124	140	125	136	129	135
136	137	124	132	124	120	127	135
138	137	121	129	126	136	123	123
185	139	135	136	125	130	140	122

(a) Original frame

170	-40	-55	-23	-10	3	3
-30	-4	-3	0	6	4	8
10	6	1	-1	1	5	2
-2	6	-6	0	-1	-5	3
5	2	9	2	1	2	-2
5	-5	0	-2	3	0	0
-4	2	0	3	0	2	1
0	1	2	1	0	-1	1

(b) Frame after DCT

Figure 1.2: DCT Transformation (8x8 block)

1.1.3 Quantization

Quantization reduces the number of bits needed to represent a pixel or sample by reducing its precision [2]. Quantization error is the main source of lossy compression [4] and for that reason, DCT coefficients are quantized to obtain higher compression. Though this results in data loss, the quality is only marginally impacted because most of the data loss is from higher frequency coefficients that are less noticeable to people. Quantization is obtained by dividing DCT coefficients that are obtained earlier by a quantization matrix. This matrix is manipulated by a quantizer scaler code and is adjusted by multiplying the matrix by this scalar code. Doing this creates redundancy within the 8x8 macroblock that can be further reduced later on. Rate control, explained further on, directly manipulates this scalar in order to control the bitrate. From Figure 1.3, we can see that the coefficients become much smaller and most of them are just zero. This further reduces a large number of bits used in compression.

170	-40	-55	-23	-10	3	3
-30	-4	-3	0	6	4	8
10	6	1	-1	1	5	2
-2	6	-6	0	-1	-5	3
5	2	9	2	1	2	-2
5	-5	0	-2	3	0	0
-4	2	0	3	0	2	1
0	1	2	1	0	-1	1

(a) Before Quantization

41	20	27	15	10	0	0
17	17	21	12	0	0	0
18	3	0	0	0	0	0
1	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

(b) After Quantization

Figure 1.3: DCT Coefficient Quantization

1.1.4 Zigzag Scan and Entropy Encoding

Once the coefficients have been quantized, they are converted into a one-dimensional string through Zigzag scanning, which takes advantage of how DCT coefficients are arranged after quantization. From Figure 1.3, you will notice that the majority of coefficients are zeroes aside from the upper left portion. Zigzag scanning starts with the non-zero coefficients in the top left corner and zig zags down through the frame, as illustrated in Figure 1.4. This will produce a one dimensional array that looks like this: 41, 20, 17, 18, 17, 27, 15, 21, 3, 1, 0, …, 0. Thus, Zigzag scanning groups low frequency coefficients at the beginning of a 1-D array and puts high frequency coefficients at the end. The exact form of scanning in modern encoders does not always follow a zigzag pattern, but the principle of ordering coefficients in a way that produces redundancy is still a fundamental of scanning.

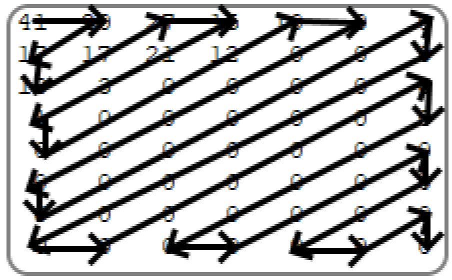


Figure 1.4: Zigzag scanning

After the Zigzag scan, the coefficients in a 1-D array are entropy coded to further reduce the number of bits in compression. The final output from a video encoder is a compressed binary bitstream which consists of *1* and *0* only. This bitstream can be either stored on disks or transmitted over networks.

1.1.5 Frame Types and Group of Pictures (GOP)

When encoding, each frame is categorized into one of three types of frames: Intra frame (I-frame), Predictive frame (P-frame), and Bi-directional frame (B-frame). P-frames and B-frames are also called inter-frames. An I-frame is treated as an independent image and coded by only applying DCT to remove spatial redundancy within the frame. It skips the motion estimation and compensation portion of video encoding entirely and is a reference frame for subsequent predictive frames.

A P-frame is coded by a forward predictive coding method, in which the difference is predicted between the current P-frame and its previously encoded I-frame or P-frame. P-frame coding exploits the similarity with previously coded frames (temporal redundancy) as well as within the frame (spatial redundancy).

A B-frame is similar to a P-frame, but it goes a step further. It is coded by exploiting the similarity with both previously coded frames and future frames as well as within the frame. Due to this, B-frames take up fewer coding bits than both I-frames and P-frames.

Both P-frame and B-frame coding are called inter-coding, which removes both temporal redundancy and spatial redundancy. I-frame coding is called intra-coding as it only removes the spatial redundancy. Generally speaking, an I-frame requires more bits to encode than a P-frame, while a P-frame needs more coding bits than a B-frame. Figure 1.5 shows each type of frame, and their respective predictive frames.

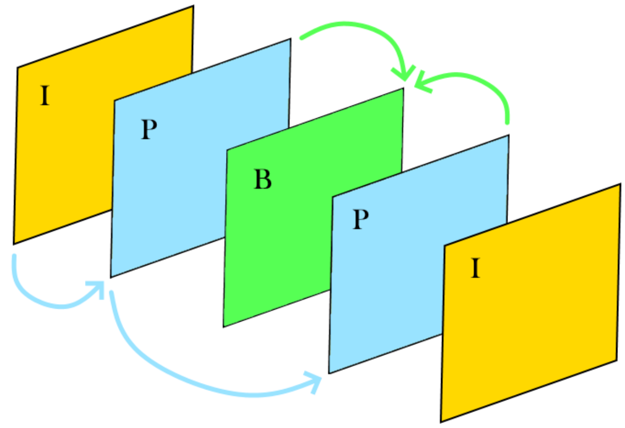


Figure 1.5: Example of each frame type and its predictive frames

A sequence of these types of frames is known as a Group of Pictures (GOP). It includes a pattern and length for each sequence which provides a structure for encoding. Each GOP starts with a reference I-frame and the length of the GOP is the number of inter-frames (P- or B-frames) between a pair of I-frames. This structure tells the encoder in what order the frames need to be encoded and allows for a variety of different and repeatable sequences. Figure 1.6 illustrates a typical GOP structure and the predictive frames each P and B frame reference [2].

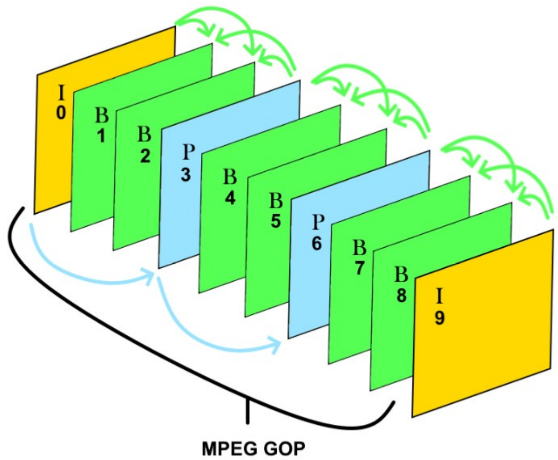


Figure 1.6: Example of a GOP

1.1.6 General Encoding Process

To end this section, we summarize the major steps used in a generic video coding system below.

Step 1. Use Motion estimation and compensation (ME & MC) to reduce temporal redundancy. If the current frame is the first frame, the encoder does not perform ME & MC since the first frame does not have the previous neighboring frame to reference. This is called intra-coding. If the current frame is not the first frame, the encoder performs ME & MC to reduce temporal redundancy and obtains the frame difference between the current frame and its neighboring reference frames.

Step 2. Apply DCT to a frame or a difference frame to reduce the spatial redundancy. The result obtained in this step is the frequency or DCT coefficients.

Step 3. Quantize DCT coefficients. Quantization reduces the precision of data and results in a loss of video signal. This step is responsible for the majority of compression seen during the video encoding process at the cost of minor visual degradation.

Step 4. Zigzag scan and entropy code the quantized coefficients to further reduce bits.

After applying the above major steps, the output of a video encoder is a compressed binary bitstream, which is suitable for storing or transmission. Figure 1.7 shows an overview of a generic video coding system.

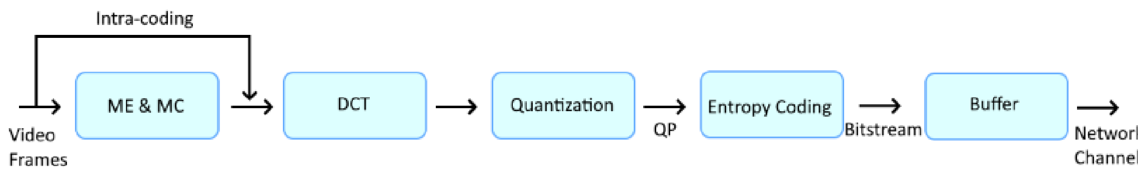


Figure 1.7: Generic Video Coding System

1.2 Video Compression International Standards - Overview

1.2.1 Compression Standard Overview

The fast-growing demand for heterogeneous video applications has stimulated the advancement of video compression technology. Over the years, different compression standards have been developed to meet the different needs for various video applications. Two international organizations have taken charge of developing video compression standards. One is the International Telecommunication Union (ITU), and the other one is the International Organization for Standardization (ISO) - Motion Picture Experts Group (MPEG).

The compression standards developed by ITU mainly target real-time video communication applications such as video conference and video phone, while ISO puts more focus on investigating standards for video storage and broadcast. In recent years, the experts from both ISO-MPEG and ITU started to work jointly to develop new compression standards for the next generation: Ultra-HDTV.

Both H.261 [5] and H.263 [6] were developed for videophone and video conferencing by ITU in 1990 and 1998 respectively. H.261 only provided low quality videophone and video conferencing over ISDN. It supports variable bit rates of nx64 Kbps (n=1..30). H.263 was developed based on H.261 and has many technical improvements. It supports low quality video phone to high quality video conferencing with bitrates ranging from 10 to 384 Kbps. H.263 has made tremendous gains in its video quality over H.261.

MPEG-1 [7] and MPEG-2 [8] were developed for video storage and broadcast by ISO in 1992 and 1994 respectively. MPEG-1 is both a video and audio compression standard. Its typical applications are MP3 and Video CD (VCD) with bitrates up to 1.5 Megabits per second (Mbit/s). To meet users' requirements for higher resolutions, MPEG-2 was designed for applications at much higher bitrates (higher than 2 Mbit/s), which were not supported by MPEG-1. As a superset of MPEG-1, MPEG-2 offers higher data rates, better error resilience and higher compression performance with its typical applications including HDTV and DVD.

Since both H.261 and H.263 were targeted for real-time video communication, they cannot be used to store videos and do not provide random access throughout a video sequence. MPEG-1 and MPEG-2 were designed for storage and broadcast, but they are not optimized for network transmission. There was no universal standard for both video storage and transmission. Therefore, there was a need to develop a new compression standard which could work well for storage, broadcast, and network transmission.

To address this need, MPEG-4 [9] was developed by ISO following MPEG-2. It is a standard for multimedia applications that aims to provide a solution to all applications in the fields of computer, telecommunications, and TV/file industries. It covers a wide range of multimedia applications, with bit rates as low as 5 Kbit/s and as high as 5Mbit/s. It first adopts object-based coding in which each object is encoded and transmitted separately. When network bandwidths are limited, important objects can be coded with higher bitrates to obtain better quality, while non-important objects can be coded with lesser quality and lower bitrates, or even discarded entirely, in order to save bits. By doing this, the coding efficiency is improved. The object-based encoding also introduces content interaction since people can manipulate objects in a scene or a frame. When compared to the previous standards, MPEG-4 demonstrates higher compression efficiency and offers flexible content-based interactivity.

With the objective to further improve compression performance and provide better error resilience, the experts from ITU and ISO formed a Joint Video Team (JVT) and jointly developed a new compression standard - H.264/AVC [10]. The ITU call it H.264 while it is known as MPEG-4 Advanced Video Coding (AVC) by the ISO. Some new techniques have been developed in this new standard including Rate-Distortion Optimized (RDO) motion estimation and mode decision, flexible block size, Intra frame prediction, and multi-frame motion estimation. Correspondingly, H.264/AVC provides 30%~50% better compression than MPEG-2, and around 30% over MPEG-4 and H.263. H.264/AVC has a wide variety of applications ranging from mobile services and video streaming, to D-Cinema, HDTV, and HD video storage.

High Efficiency Video Coding (H.265/HEVC) [1] is the most recent video coding standard developed jointly by ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Moving Picture Experts Group (MPEG) who previously worked together on the H.264/AVC standard [10]. The objective for this current standard was to further increase the compression performance over H.264/AVC and previous compression standards. H.265/HEVC proposes a number of new techniques in order to obtain higher compression performance. While H.265/HEVC follows most of the same conventions that were presented in H.264/AVC, such as inter and intra picture prediction, and similar entropy encoding techniques, it has optimized and improved these techniques while also introducing new technology that makes the standard more flexible than its predecessor. A key feature of this new standard is the redesign of macroblocks from the previous standard. In previous versions, a macroblock would consist of a 16x16 block containing the luma information with two corresponding 8x8 chroma blocks [1]. This has been changed in H.265/HEVC to instead use coding tree units (CTU) which is similar in structure to macroblocks but allows for luma blocks and their corresponding components to be either 16x16, 32x32, or 64x64 [1]. This offers a high level of granularity and optimization, with frames being broken down according to their complexity, thus encoding them more efficiently than previous encoders.

Using new compression techniques, H.265/HEVC was able to achieve over 50% increase in compression efficiency when compared to the previous standard H.264/AVC. It also supports resolutions up to 8K Ultra-high-definition TV (8192x4320). Our research work is based on this current compression standard (H.265/HEVC).

1.3 Video Rate Control

1.3.1 Rate Control Definition

Due to the properties of entropy coding and the nature of content change among video frames, video encoding produces a compressed bit stream with variable encoding bitrates [11]. There is an issue when transmitting compressed bitstreams over constrained network channels. On one hand, network congestion and video data loss might be caused if the encoding bitrate is larger than the network bandwidth. On the other hand, video coding quality might be degraded unnecessarily, and network bandwidth might be wasted if the encoding bitrate is less than the channel bandwidth. To resolve this issue, rate control (RC) must then be adopted to decide the encoding parameters to regulate the encoding bitrate of a video that is being transmitted over a network within limited bandwidth. At the same time, it ensures the optimal amount of bandwidth is being utilized to maximize the encoding quality [12]. Since channel bandwidths are limited and time-varying, rate control is a must to achieve the best tradeoff between encoding quality and bandwidth utilization. Without rate control, it is impossible to have various video applications for either storage or transmission. Therefore, it plays a crucial role in video compression and transmission.

Two different network channels types exist: Variable Bit Rate (VBR) and Constant Bit Rate (CBR) [13]. CBR requires that the bitrate over the channel be constant. A buffer is used to smooth out any variation in bitrate that can occur within a video in order to maintain a constant bitrate. Instead of requiring a specific bitrate to be maintained, VBR provides a set of parameters that must be followed [13]. VBR is expected to maintain a certain level of quality or adapt as needed based on network conditions and, as such, is not as strict as CBR. Rate control considers both types of channels and adjusts the bitrate accordingly to fully utilize the available bandwidth.

1.3.2 Rate Control Basics

Generally speaking, rate control consists of several core components: target bit allocation, buffer control, and rate-distortion modeling. We will discuss them, along with rate control steps, in the section below.

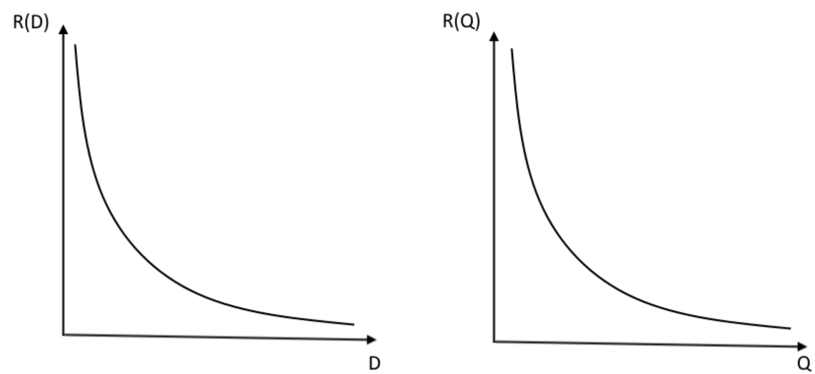
1. Target bit allocation

Target bit allocation focuses on how to estimate the target bit budget to encode a coding unit, i.e. a frame or a macroblock (MB). Given the target encoding bitrate, which is dependent on the current available network bandwidth, the estimation is based on various information such as the available remaining bits, the coding unit's complexity, and the number of remaining coding units to be encoded.

2. Rate - Distortion Model and Rate - Quantization Model

The Rate-Distortion (R-D) model, or the Rate-Quantization (R-Q) model, plays a key role in RC [14]. Rate (R) represents the target coding bitrate, which is normally given by the current available network bandwidth while distortion (D) indicates the reconstructed error caused by compression. Distortion is in inverse relation to coding quality [15]. The higher the distortion is, the lower the encoding quality. The encoding parameter, used in controlling encoding rate and distortion during the coding process, is called the quantizer (Q) or quantization parameter (QP). The Q determines how much of the encoding frame will be quantized and has a direct effect on the distortion of this frame. Rate control manages the bitrate of a video by directly modifying Q [13].

The R-D model and the R-Q model, as illustrated in Figure 1.8, characterize the relations among the target coding bit rate, distortion, and the quantizer (Q) for video coding. From Figure 1.8 (a), we can find that distortion and rate have an inverse relationship, i.e., a decreasing D leads to an increasing R and an increasing D is caused by a decreasing R. So, the R-D model is a tradeoff between R & D. We can observe from Figure 1.8 (b) that R decreases when Q increases and vice versa. Similarly, the R-Q model also represents a tradeoff between R & Q. By combining these two models, we can draw the conclusion: when Q increases, R will decrease, but D will increase, which results in a higher compression performance with a lower encoding quality. On the contrary, when Q decreases, R will increase but D will decrease correspondingly. This will reduce compression performance and improve the coding quality.



(a) R - D Model (b) R - Q Model

Figure 1.8: Rate-Distortion model and Rate-QP model

Given the target bitrate R, which is the available channel bandwidth, we can calculate the Q according to the R-Q model and then use this Q to control the actual encoding rate to be close to the given target bitrate. If a R-D or a R-Q model is precise enough, rate control can be achieved accurately. Therefore, we can tell how important these models are to rate control.

3. Buffer in Rate Control

Traditional rate controllers normally place a buffer between an encoder and a network channel to smooth out the rate variation. The encoder compresses a frame and sends the encoded bits to the buffer, while the channel takes out the compressed bits from the buffer and transmits them over a network. If the output encoding rate from a video encoder exceeds the channel bandwidth, the buffer stores the excess encoding bits to reduce bitrate fluctuation. The goal of buffer control is to maintain buffer fullness at a certain level so as to avoid buffer overflow and underflow. Buffer overflow will cause frame data loss and interrupt video smoothness; while buffer underflow will waste channel bandwidth and degrade the coding quality unnecessarily. Thus, buffer fullness should be maintained at a certain level in order to avoid buffer overflow and underflow.

The use of a buffer in rate control also brings in an amount of delay in transmission. A larger buffer corresponds to a longer transmission delay, while a smaller buffer provides low delay at the risk of frame data loss. A buffer's size is normally decided according to the delay permitted by various applications [13].

The core methodologies for controlling a quantizer have centered around deriving efficient buffer control algorithms such as [16]. Such methods focus around the output buffer that encoded frames are loaded into before being transmitted. In CBR scenarios, the output buffer will have a set, constant bitrate wherein the optimal amount of buffer fullness is at 50% utilization. This means that the amount being transmitted matches the amount being loaded into the buffer. While various implementations have been proposed, the underlying principle is the same with buffer-focused algorithms: monitor buffer fullness and adjust the quantization parameter accordingly. If the buffer is starting to overflow, then the QP is increased to further compress the incoming frames. Thus, fewer and fewer coding bits are sent to the buffer to lower the buffer fullness. If the buffer is underflowing, QP is decreased gradually, which generates more and more encoding bits so as to increase the buffer fullness correspondingly. Figure 1.9 demonstrates a high-level view of a buffer in rate control.

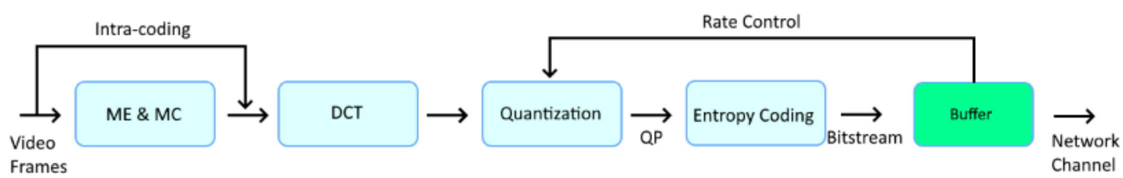


Figure 1.9: Buffer in Rate Control

1.3.3 Rate Control Process

After introducing the fundamental concepts of rate control, we describe the common rate control process in a stepwise manner. By adjusting the QP of a coding unit, rate control can directly affect the coding quality and regulate the coding rate. Assume that the target encoding bitrate, determined by available network bandwidth, is already known before encoding.

Step 1. According to the current available information, such as bandwidth, remaining available bits, remaining coding units, and coding complexities, estimate the target bit budget to encode the current coding unit.

Step 2. Based on buffer fullness, the number of target bits estimated is further adjusted to avoid buffer overflow or underflow.

Step 3. Given the target bits, compute the encoding parameter, QP, according to the R-Q model.

Step 4. Use the QP to encode the current coding unit. If the R-Q model is accurate enough after encoding, the number of actual encoding bits should be very close to the number of estimated target bits.

Step 5. According to the coding results, dynamically update the R-Q model to make it become more and more accurate to reflect the characteristics of the coding sequence. If the buffer tends to overflow, skip encoding the next coding unit.

Step 6. Go to Step 1 to encode the next coding unit until the end.

After applying the above steps to every coding unit, the actual encoding rate should be close to the target encoding rate.

CHAPTER 2 LITERATURE REVIEW

Rate control has been an important research topic in literature and well studied over the years. Various rate control algorithms have been developed for various applications and compression standards. In this section, we will review typical rate control algorithms and related work.

2.1 Rate Control in International Video Compression Standards

Throughout the history of video encoding, rate control has been an important part of ensuring that encoding is performed smoothly and does not go over its predetermined limits despite not being a normative part of any encoders. Features such as rate control are not usually specified for the encoder which allows for different algorithms and implementations to be proposed and adopted. Rate control has evolved over the years and looking back at previous rate control methods for different compression standards provides us a better foundation to understand newer rate control technology.

2.1.1 H.261 Rate Control

The first of ITU's (International Telecom Union) H.261 codecs designed primarily for video conferencing [13, 17]. This encoder introduces concepts to video encoding that are used in modern encoders such as DCT, motion estimation, and motion prediction [17]. Encoding is done primarily through DCT, quantization, and prediction [17].

Due to video conferencing mainly only having to deal with the upper half portion of people's bodies and low movement, the rate control in this standard simply keeps track of the buffer fullness and makes adjustments to the QP accordingly [13]. It primarily controls the encoded bits by modifying the QP of the frame being encoded based on the amount of buffer fullness. Along with modifying the QP, if buffer fullness hits a critical threshold, the rate control algorithm for this encoder will not send data to prevent further overflow [13]. On the other hand, if the buffer reaches a critical underflow threshold, then it will add additional compressed bits to maintain a balanced buffer fullness [13].

2.1.2 MPEG-1 Rate Control

MPEG-1 is the first standard developed by the Motion Picture Experts Group (MPEG) and was designed for storage media such as CD-ROM [7]. MPEG-1 uses Group of Pictures (GOP) which consists of three different types of frames: I, P, and B. The rate control of MPEG-1 monitors buffer fullness similar to H.261, considers the GOP structure, and distributes the number of target bits based on the GOP size and frame type to be encoded. P frames should obtain 2-5x the amount of bits compared to B frames, and I frames should receive 3x the amount of bits compared to P frames.

2.1.3 MPEG-2 Rate Control

MPEG-2 is a continuation of the MPEG-1 format designed for higher resolution formats such as DVD and HDTV [18, 19]. It is a superset of the MPEG-1 standard and is backwards compatible meaning it can decode any MPEG-1 video [18, 20]. The MPEG-2 rate control consists of three main steps to regulate bitrate [8, 13, 21]. First, it calculates the number of bits that should be allocated to a GOP sequence. This takes into consideration the number and types of frames being encoded, along with any error in encoding previous GOP sequences [21]. Then, it estimates the target bits for a given frame. This is done through taking into consideration the type of frame (I, P or B) being encoded, the frame complexity, along with the number of bits that are left for that particular GOP sequence. Once the target bit rate is found, based on a linear R-D model, MPEG-2 rate control computes the QP for each macroblock. Finally, the QP is adaptively adjusted based on the complexity of the macroblock and is used to encode this macroblock.

2.1.4 H.263 Rate Control

Based on H.261, H.263 was developed in 1995 to handle low bitrate communication [17]. The rate control for H.263 allocates target bits to a frame according to the channel bandwidth and frame rate, without considering frame complexity, numbers of frames left to code, number of bits already spent in coding, etc. [22]. According to the buffer status, the frame-level target bits are adjusted linearly to avoid buffer overflow or underflow. If the buffer is too full, exceeding a certain threshold, then the frame is skipped for encoding to reduce buffer fullness. H.263 rate control is performed at the macroblock level, but not the frame level. The target bits for a frame are further distributed to each macroblock according to the variance in a macroblock and the bits left for the frame being encoded [22]. Once the target bits for a macroblock is determined, we can obtain the QP for this macroblock using a logarithmic R-D model and use this QP to encode the macroblock.

2.1.5 MPEG-4 Rate Control (SRC)

The rate control adopted in MPEG-4 is scalable since it can work at the frame level, macroblock level, and object level [9, 23]. MPEG-4 rate control sets the buffer size in regard to the latency requirements of applications and initializes the target buffer fullness level to half of the buffer size. The target bit budget per frame is estimated according to the actual encoding bits used in coding the previous frame and average remaining bits available per frame. Then, the frame target bit budget is modified linearly according to buffer occupancy to maintain buffer fullness at the target level. The quadratic rate-quantizer model below was first introduced in MPEG-4 rate control and has become one of the leading models. The QP is calculated for the frame being encoded based on the quadratic rate-quantizer model [23]:

$$R = \frac{X_1 \times S}{Q} + \frac{X_2 \times S}{Q^2},$$

where X_1 and X_2 are model parameters that are updated dynamically after encoding each frame, S represents the frame complexity which is based on the Mean Absolute Difference (MAD) of the frame, Q is the quantizer, and R is the number of target bits estimated for the frame.

The advantage of this model is its accuracy and simplicity. It has very low computational complexity and can quickly adapt to reflect the characteristics of the coding sequence. Therefore, this quadratic R-Q model has been widely used in rate control since then. Since X_1, X_2, and S are already known, once R is obtained after target bit estimation and buffer control, Q can be calculated based on this quadratic model and is then used to encode the current frame. After encoding, X_1 and X_2 are updated based on the actual encoding bits and QPs for previous frames. This can make the R-D model become more and more accurate to represent the R-D characteristic of the coding sequence over time. If the buffer fullness is over 80% of buffer size, the encoder skips the coding of the next frame.

2.1.6 H.264/AVC Rate Control

Rate control in H.264/AVC also adopts the above conventional quadratic Rate-Quantization (R-Q) model proposed for MPEG-4 to determine QP [24, 25]. However, since a new coding technique, Rate Distortion Optimization (RDO), is adopted in H.264/AVC, the MAD of the current inter coding unit (P&B) used in this R-Q model cannot be obtained. To resolve this problem, by using a linear MAD prediction model, H.264 rate control predicts the MAD of the current coding unit based on that of the previous encoded coding unit. To obtain fine-grain rate control, H.264/AVC also introduces the *basic coding unit*, which is either a frame, slice, or a set of MBs. This can achieve fine-grain rate control and provide high accuracy, but at the cost high complexity. H.264 rate control first estimates the total number of target bits allocated to each GOP and sets the initial QP of the 1st frame in each GOP. Then, it uses a linear model to predict MAD of the current basic unit and adopts the same quadratic R-Q model to calculate the QP to be used for RDO in this basic unit.

2.1.7 H.265/HEVC Rate Control

A rate control algorithm was proposed using a pixel wise unified rate-quantization model for H.265/HEVC [26]. This R-Q model is almost the same as the conventional quadratic R-D model with little variation. Due to the structure of Coding Tree Units (CTU) and their corresponding CU subunits, the encoder will partition Coding Units (CUs) if available and if splitting results in a smaller sum RD cost when compared to the RD cost of the parent CU. Effectively, each CU within a CTU will have the most optimal RD cost possible [26]. Because of this, each CU could potentially have their own R-D model, and thus computing the QP would be computationally expensive considering that a frame could have any number of CUs. To save on computation time, a single unified R-D model was proposed in [26] that could be applied to all CUs within a frame, which can save computation time and reduce implementation complexity. The proposed unified quadratic R - D model [26] for each different CU size is described as:

$$\frac{R[k]}{N_{pixels}[k]} = MAD_p[k] \times \left(\frac{X_1[k]}{Q_\delta^2[k]} + \frac{X_2[k]}{Q_\delta[k]}\right),$$

where R[k] describes the target bits for either a CTU or a frame, $N_{pixels}[k]$is the number of pixels for the frame or CTU, $MAD_p[k]$is the predicted distortion, Q is the quantization parameter, $X_1[k]$ and $X_2[k]$are model parameters that are updated after encoding [26].

To calculate the QP using this model, the target bits are first estimated by finding the bit budget for the frame being encoded. Before the bit budget for a frame can be found, the bit budget for the GOP must be determined. The bit budget for the GOP is calculated by dividing the bitrate over the framerate and then multiplying by the number of frames in the GOP [26]. This is then further adjusted by checking the buffer state to see if it has too many or too few bits. A frame's bit budget can then be determined based on the available bits left for that GOP and what type of frame is being encoded. The final target bits are then adjusted based on the buffer status. The initial target bits are reduced accordingly if the buffer tends to overflow; otherwise, more target bits will be assigned if the buffer is starting to underflow. This model and rate control scheme were first adopted into version HM8.0 by H.265/HEVC.

2.2 Recent Advances in Rate Control for H.265/HEVC

With the advent of H.265/HEVC, a number of rate control algorithms have been proposed recently [26-40]. As we discussed in detail in Section 2.1.7, the research in [26, 27] observed that the ratios of distortions over bits for all the blocks are nearly constant due to using rate distortion optimization (RDO) technique. Therefore, based on the characteristics of similar ratios of distortions over bits regardless of block sizes, this work proposed a pixel-based unified Rate-Quantization (R-Q) model and developed a rate control (RC) algorithm. Due to its excellent performance, this work was adopted by HEVC reference software.

Li et al. [28, 29] proposed an R –λ model, where λ is the Lagrange Multiplier based on a frame's complexity and then developed a rate control (RC) algorithm based on this model for H.265/HEVC. The algorithm allocates target bits to a GOP, a frame, or a Coding Tree Unit (CTU) after which it linearly adjusts the target bits based on buffer status. Due to its excellent RC performance, this work was integrated by Joint Collaborative Team on Video Coding (JCT-VC) to be part of the HEVC Test Model [29]. Since this work uses the optimization technique for determining encoding parameters so as to achieve accurate rate regulation, it induces relatively high computational complexity in addition to the already very complicated HEVC encoding process.

After this work, a few other R – λ based rate control algorithms have also been recently proposed. The researchers in [30] proposed a gradient-based R – λ model and inner-frame rate control for HEVC. This work adopts a gradient per pixel to measure a frame's complexity, calculates Lagrange Multiplier, and assigns target bits to improve RC performance. In order to enhance facial details for video conferencing, the work in [31] developed a RC algorithm based on a weighted R – λ model which can allocate more target bits to important regions in a video frame. To control video quality fluctuations, the researchers in [32] established a model between the distortion and λ. According to buffer fullness, it adjusts the CTU level λ and target bits to meet channel bandwidth for video communications.

The research in [33, 34] developed typical optimization rate control algorithms at the frame level and the block level for H.265/HEVC. To achieve satisfactory coding performance for intra coding, a Structural SIMilarity (SSIM) – based game theory approach was introduced in [33] to optimize the CTU-level bit allocation via the Nash bargaining solution. A ρ– domain bit allocation and a rate control algorithm have been proposed in [34] to optimize bit allocation for key frames [34]. They established an effective model for transformed coefficients and exploited skip-model CU percentage to evaluate frame-level quality dependency. Zhou et al. developed a SSIM-based RC scheme to achieve optimal CTU level bit allocation given the bitrate budget [35]. They established a SSIM-based R-D model and solved the global optimization problem using convex optimization.

An optimal bit allocation (OBT) for CTU-level rate control was introduced in [36]. It built an R-D estimation, formulated optimal bit allocation, and used a recursive Taylor expansion (RTE) method to obtain an approximate solution. Based on content complexity correlation, the researchers in [37] developed a CTU level rate control for HEVC. The frame level rate control scheme proposed for x265 in [38] not only makes use of coding complexity history, but also considers data dependencies between the current and near future frames. To improve rate regulation performance, the work in [39] investigated a learning-based method for the optimal initial quantization parameter (QP) prediction. To enhance the prediction accuracy of a $R - \lambda$ model, the research [40] investigates a joint machine learning and game theory method to build multiple CTU-level models for inter frame RC. The model selection of a CU depends on the support vector-machine and Nash-bargaining-solution.

In summary, the aforementioned algorithms adopt different approaches to implement RC for H.265/HEVC, aiming to achieve excellent RC performance. These algorithms and other conventional RC algorithms have one common property: they typically adopt a buffer in rate control to smooth out encoding bits and to linearly adjust frame target bits according to buffer status. According to the best of our knowledge, all current H.265/HEVC rate control algorithms traditionally employ buffers in implementing rate regulation.

A Proportional-Integral-Derivative (PID) controller is a feedback control mechanism that is widely used in industrial control systems and various other applications [41]. We were the first to introduce this controller into video compression and used it in controlling buffer fullness effectively [42, 43, 44, 45]. In [46], we proposed a new way to use it directly in controlling encoding bits for MPEG-4 rate control, without using and controlling a buffer. Since a buffer is not used in rate control, our algorithm improves the real-time response and reduces encoding delay.

CHAPTER 3 RESEARCH CHALLENGES AND OVERVIEW OF THIS WORK

3.1 Research Challenges

Due to its high coding efficiency, H.265/HEVC has a wide range of applications and has received extensive attention recently. To achieve high compression performance, H.265/HEVC has evolved from previous video coding standards by adding more advanced features and a number of new coding techniques such as advanced motion vector prediction, quarter sample precision for motion compensation, parallel processing architectures, diverse coding block sizes, and coding structures. With all these new features, the encoding scheme of H.265/HEVC is much more complicated than previous standards. As a result, H.265/HEVC rate control has become much more sophisticated and challenging. Furthermore, due to high demand for real-time networked video applications, how to compress videos and regulate encoding rates in real-time with low delay remains as another research challenge.

Due to these changes, the complexity of H.265/HEVC compression is very high when compared with previous standards. The majority of HEVC's computational complexity is caused by many possible diverse block sizes and coding modes [26]. Therefore, a practical rate control scheme for HEVC should be relatively simple for many commercial video applications [26].

3.2 Overview of This Work

In order to address the challenges listed above, in this study, we conducted in-depth research on rate control for H.265/HEVC. There were two research objectives. The first one was to achieve accurate rate control and the second one was to improve real time response and reduce encoding delay.

According to the best of our knowledge, so far, the Proportional-Integral-Derivative (PID) control theory has not been used in directly controlling encoding bitrates without using a buffer for H.265/HEVC. Thus, based on the PID control theory and the new coding features of H.265/HEVC, we developed a new PID bit controller for H.265/HEVC to directly control compression bitrate without using a buffer. We first performed target bit allocation to a coding unit according to the available channel bandwidth. Then, based on the bit error between target bits and actual encoding bits, we applied PID control theory to adjust the target bits allocated to a coding unit proportionally, integrally, and derivatively, aiming at reducing the bit error and achieving accurate rate control.

The innovation of the algorithm is that the buffer is not used in the entire rate control process and compression rates are directly controlled by applying the famous PID theory in the field of Automatic Control. The compressed bits are immediately sent to network channels for transmission and are smoothed by network buffers if needed. When combined with our simple bit allocation strategies, our rate control algorithm is not only effective, but it also reduces encoding delay and improves real-time response. Experimental results have authenticated the effectiveness of our rate control algorithm.

The rest of this thesis is organized as follows. Chapter 4 discusses the direct bit control, and Chapter 5 investigates the bufferless rate control algorithm in detail. Chapter 6 explains the development of automation tools for collecting experimental data. Chapter 7 presents the experimental results to demonstrate the performance of the algorithm. Finally, Chapter 8 concludes this thesis and explores possible future work.

CHAPTER 4 DIRECT BIT CONTROL FOR H.265/HEVC

4.1 Direct PID-Based Bit Control for Inter-Frames

The Proportional-Integral-Derivative (PID) controller is a feedback control loop to regulate the actual output from a process to be close to the target output. It integrates a proportional term, integral term, and derivative term to achieve accurate and stable control. Due to its flexibility and broad range of applications, it is a common control system found in various industries and control applications [47].

Since rate control is a process that is automatically controlled, it is a suitable candidate for PID Control. When a frame is encoded, the goal is to regulate the number of actual encoding bits output from an encoder to be close to the target bit budget allocated to a frame before encoding. At the beginning of the encoding process, the difference between actual encoding bits and frame target bits might be very large due to the natural characteristics of the prediction from a rate-distortion model. But through the model's self-adaptation during the encoding process, the generic R-D model gradually adapts to reflect the R-D properties of an encoding sequence, and thus the bit difference gradually diminishes. By applying this bit control for each frame during the encoding process, we can achieve rate control.

Conventional rate control approaches, including RC for HEVC, generally use a linear method to adjust target bits for a coding unit according to buffer status. That is to say that they only use a proportional controller in rate control. The weakness of a proportional controller is that it can only reduce the control error, but can never eliminate this error [41]. Obviously, its control ability is not effective enough. As the effectiveness of applying the PID controller in video compression and rate control has already been demonstrated [42, 43, 44, 45, 46], in this research, we further investigate on using the PID technique to control encoding bits directly for H.265/HEVC inter-frames.

In accordance with available bandwidth, rate control allocates a number of target bits to a given frame and, once encoded, collects the actual number of encoding bits used. The control target is to maintain the number of actual encoding bits per frame closely to its allocated target bit budget, and to minimize the difference between target bits and actual encoding bits for a frame in order to achieve accurate rate control.

We define the error value *e(t)* as the difference between the number of target bits $T_{i,j}$ allocated to the j^{th} frame in the i^{th} GOP, and the number of actual encoding bits $C_{i,j}$ used in encoding this frame:

$$e(t) = T_{i,j} - C_{i,j} \tag{1}$$

An illustration of each portion of the PID controller is shown in Figure 4.1. Once a frame has been encoded, a PID controller continuously calculates this error value *e(t)*, and then applies a correction based on proportional, integral, and derivative terms. The controller attempts to minimize the error during the entire encoding process by applying a control variable $\mu(t)_{PID}$, representing the PID bit adjustment, to the target bits pre-allocated to a frame before encoding [48].

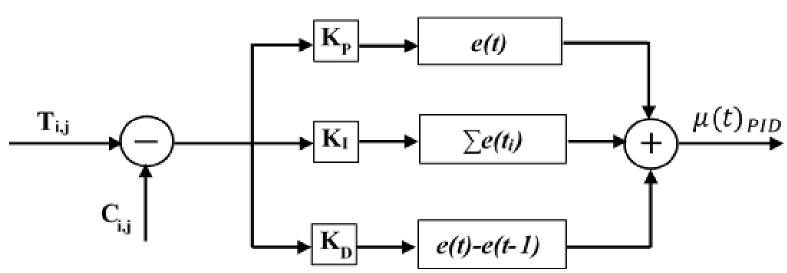


Figure 4.1: The PID-Based Bit Controller

The control function of the PID bit adjustment is expressed by:

$$\mu(t)_{PID} = \mu(t)_P + \mu(t)_I + \mu(t)_D, \quad (2)$$

where $u(t)_P$ is the proportional component, $u(t)_I$ denotes the integral portion, and $u(t)_D$ represents the differential term. Specifically, these three components can be calculated according to:

$$\mu(t)_P = K_P \cdot e(t),$$

$$\mu(t)_I = K_I \cdot \int_0^t e(t')dt',$$

$$\mu(t)_D = K_D \cdot \frac{de(t)}{dt},$$

where K_P, K_I, and K_D are positive and represent the control parameters for the proportional, integral, and derivative terms respectively. *t* is the present time, while *t'* is the integral from time *0* to the present *t*.

Each of these components contributes to reducing the error found while encoding. The proportional component ($\mu(t)_P$) corrects the control target proportional to the difference e_t between the actual bits and the target bits per frame. It can reduce the difference, but never eliminate this difference. Most rate control approaches only use proportional control alone. However, this will lead to an error between the target bits and the actual encoding bits since it requires an error to produce the proportional response. Otherwise, there is no corrective response if there is no error. When the residual error induced by the proportional control lasts, the integral term in ($\mu(t)_I$) can gradually increase the control strength due to the growing integral effect and can eliminate this error. However, doing so may result in slow response and overshoot. The derivative term ($\mu(t)_D$) considers the rate of error change, but not the error itself. It can reduce overshoot, improve response, and ensure system stability [48]. Thus, using the three controllers simultaneously is much more powerful than using each individual controller since it integrates their benefits together.

As video encoding is performed on a frame by frame basis, it is in a discrete form but not in a continuous form. Therefore, $\mu(t)_I$ and $\mu(t)_D$ are actually implemented by the following discrete forms:

$$\mu(t)_I = K_I \cdot \int_0^t e(t')dt' = K_I \cdot \sum_{i=1}^{t} e(t_i),$$

$$\mu(t)_D = K_D \cdot \frac{de(t)}{dt} \approx K_D \cdot (e(t) - e(t-1))$$

4.2 PID Control Parameter Tuning

PID Tuning is a repetitive process which adjusts its control parameters to optimum values, so as to obtain desired control objectives. During the tuning process, system stability is a fundamental requirement, but different systems and applications might have different behaviors and requirements. Therefore, parameter tuning methods are different. Ziegler–Nichols, software tools, Cohen–Conn, and manual tuning are general approaches for parameter tuning [41].

Most control systems empirically select these PID parameters. A larger proportional coefficient K_P will have a stronger control power, but might also cause system oscillation. K_i can eliminate the error, but too much K_i will cause instability. Even though K_d is able to reduce overshoot, improve response, and ensure system stability, too much K_d will cause excessive response and overshoot. Therefore, how to select appropriate parameters is a tough and very time-consuming process.

In this study, we manually tuned these three coefficients. Based on our experimental results, we found that if K_p, K_i, and K_d are around 0.3, 0.025 and 0.1 respectively within a small range, our rate control algorithm can obtain very good performance. This indicates that our PID controller is very stable and is not that sensitive to these values. Please note that these three values are generic and work for all video sequences at all target bit rates.

CHAPTER 5 THE PROPOSED BUFFERLESS RATE CONTROL FOR H.265/HEVC

In this chapter, we first present our proposed algorithm in-depth, and then summarize the algorithm in a stepwise manner.

5.1 Target Bit Allocation

5.1.1 Bit Determination for H.265/HEVC Rate Control

The four bit determination cases in the H.265/HEVC rate control (RC) algorithm are described below:

1) The 1^{st} frame in the first GOP: H.265/HEVC RC directly uses the pre-set QP value in the encoding configuration file.
2) The 1^{st} frame in other GOPs: H.265/HEVC RC does not perform target bit allocation nor does it calculate its QP through the pixel-wise R-Q model. Instead, its QP is directly obtained based on the QP of the last frame in the previous GOP and the average QP of reference frames from the previous GOP. Then, this QP is clipped within ±2 of the QP of the last frame in the previous GOP, to guarantee smoothness.
3) Reference frames in a GOP: Multiple factors have been exploited in allocating the target bit budget to reference frames, including buffer occupancy, target buffer level, weights of reference and non-reference frames, remaining bits in a GOP, and the initial buffer status. Once the number of target bits of this reference frame is obtained, the QP for this frame can be derived by the pixel-wise unified R-Q model.

4) Non-reference frames in a GOP: H.265/HEVC RC does not allocate target bits to non-reference frames nor compute its QP through the pixel-wise R-Q model. Instead, for a non-reference frame, HEVC RC directly derives its QP by adding 2 to the average QP of its two reference frames.

In summary, H.265/HEVC RC only performs target bit allocation to the reference frames in a GOP, but not for non-reference frames and the 1st frame in a GOP. The bit allocation distributes target bits to a frame according to the remaining bits in a GOP, and the weights of reference and non-reference frames, then proportionally adjusts the target bit budget according to buffer feedback. Once the target bit budget for this reference frame is obtained, the QP for this frame can be derived by the pixel-wise unified R-Q model [26].

5.1.2 The Proposed Bit Allocation Strategy

The strategy for allocating the initial target bits for each frame can be broken down as follows:

1) The 1st I-frame, P-frame, and B-frame: we do not perform target bit allocation nor calculate QP for them. Instead, we use the pre-defined QP in the encoding configuration file to encode these frames.
2) I-frame for each intra-period: Same as H.265/HEVC RC, we do not allocate target bits for intra frames, but we use our proposed method to calculate the QP for each I-frame.
3) Bit Allocation for Inter–frames (P-frames and B-frames): Different from H.265/HEVC RC, which only allocate bits for P-frames (reference frames) but not for B-frames (non-reference frames), our strategy is to perform bit allocation for both P-frames and B-

frames. Here, we propose a simple but effective bit allocation method. We first allocate the target bits at the GOP level and then allocate the target bits at the frame level.

Our method for estimating the number of bits to allocate a GOP is defined as:

$$T_{GOP} = \frac{T_{rate}}{F_{rate}} \times N_{GOP}, \tag{3}$$

where T_{GOP} denotes the target bits allocated to a GOP, F_{rate} is the frame rate, N_{GOP} indicates the number of pictures in a GOP, and T_{rate} represents the target bitrate or the current available bitrate determined by the channel bandwidth. Please note that this work mainly focuses on Constant Bitrate Control (CBR) applications, but it also works for Variable Bitrate Control (VBR) cases. Once we obtain the target bit budget for a GOP, we can calculate the initial target bit budget for a frame by:

$$T_{i,j} = \frac{W_{i,j}^{S} \times T_{GOP}}{(W_{i,j}^{I} \times N_{I,i} + W_{i,j}^{P} \times N_{P,i} + W_{i,j}^{B} \times N_{B,i})} \tag{4}$$

where $W_{i,j}^{I}$ is I-frames' weight, $W_{i,j}^{P}$ is P-frames' weight, and $W_{i,j}^{B}$ is B-frames' weight. $W_{i,j}^{S}$ is $W_{i,j}^{I}$, $W_{i,j}^{P}$, or $W_{i,j}^{B}$ depending on the current frame type. Correspondingly, $N_{I,i}$, $N_{P,i}$, and $N_{B,i}$ denote the number of I-frames, P-frames, and B-frames within the i^{th} GOP. It can be seen from Eq. (4) that the proposed bit allocation method only depends on the target bitrate, frame type weights, frame rate, GOP size, and the number of I-, P- and B-frames in a GOP. This information is already available before encoding, and thus our bit allocation method is suitable for real-time control. Please note that our method also performs target bit allocation to B-frames. This is different from H.265/HEVC RC, which does not allocate target bits nor calculate QP for B-frames. Instead, H.265/HEVC RC directly derives a B-frame's QP based on its two reference QPs, which might degrade the rate control accuracy.

Once a frame has been encoded, the weight for this frame needs to be updated dynamically according to its frame type. The only frame weight that is not adjusted is the P frame which acts as our reference point and is set to a value of 1.0. Only the weights of I- and P- frames are dynamically updated. A weight adjustment method was proposed in [45], which takes the average number of encoded bits and the average distortion into consideration when making adjustments. Here, we apply this method to adjust the weights below:

$$W_{i,j}^{I} = \frac{e^{((PSNR_{i,j}^{P} - PSNR_{i,j}^{I})/\delta)} \times B_{i,j}^{I}}{B_{i,j}^{P}}, \tag{5}$$

$$W_{i,j}^{B} = \frac{e^{((PSNR_{i,j}^{P} - PSNR_{i,j}^{B})/\delta)} \times B_{i,j}^{B}}{B_{i,j}^{P}}, \tag{6}$$

where $PSNR_{i,j}^{B}$, $PSNR_{i,j}^{P}$, and $PSNR_{i,j}^{I}$ are the average PSNRs in coding previous num_B *B*-frames, num_P *P*-frames, and num_I *I*-frames respectively, and $B_{i,j}^{B}$, $B_{i,j}^{P}$, and $B_{i,j}^{I}$ are their corresponding average coding bits. For the intra-period -1, which means that only the first frame is an I-frame, and the rest of the frames are P- or B-frames, the window size (num_I + num_P + num_B) is empirically set to 30 [42]. Otherwise, the window size is twice the length of the intra period. δ is a tuning factor and is empirically set to 16. The initial values of $W_{i,j}^{I}$, $W_{i,j}^{P}$, and $W_{i,j}^{B}$ are 3.0, 1.0, and 0.5 respectively.

The weight adjustment method takes the average actual encoding bits and the average coding qualities into consideration. Its objective is to keep the coding quality consistent among different frames and types. When compared with the average coding quality of previous P-frames, if that of previous *B*-frames is lower, $W_{i,j}^{B}$ is increased and more target bits will be allocated to subsequent *B*-frames. This way, its coding quality is gradually improved. On the contrary, if the average PSNR of B-frames is higher than that of P-frames, $W_{i,j}^{B}$ is decreased and fewer target bits will be distributed to subsequent B-frames, and thus its coding quality is decreased correspondingly.

5.2 PID-Based Target Bit Adjustment

Conventional rate control schemes generally use a buffer to smooth encoding bits and proportionally tune the initial target bits for an inter-frame according to buffer status. In this study, we propose to remove the buffer from the H.265/HEVC rate control process and exploit the PID bit controller to directly manage encoding rates to meet the constraints of available channel bandwidth. The encoded bits are sent to network channels directly and are smoothed in buffers on the network side if needed. Since a buffer is not used during the encoding process the encoding delay is reduced, and the real time response is also improved.

After obtaining the initial target number of bits for a frame, we apply the PID bit controller to tune the initial target bit budget by:

$$T_{i,j} = T_{i,j} + \mu(t)_{PID} \tag{7}$$

If the actual encoded bits from the previous frame exceed the target bits that were budgeted for it, the PID controller will adjust by decreasing the target bit budget for future frames accordingly. Conversely, when the actual encoded bits for the previous frames were less than their target bit budgets, the PID controller increases the target bit budgets for the subsequent frames. Through these adjustments, the PID controller reduces the error that has been accumulated throughout encoding and achieves an actual encoded bitrate close to the target bitrate.

As a precaution, thresholds are in place to prevent too few or too many target bits from being budgeted to an inter-frame, which may result in too high or too low of coding quality. The upper bound of the frame level target bit budget is empirically set to double the number of average bits per frame, while the lower bound is set to one fourth the number of average bits per frame [46]. This ensures that a frame can obtain a minimum encoding quality while also guaranteeing that no frame will obtain an extremely high coding quality. Specifically, our thresholds are set as follows:

$$T_{i,j} = min\{T_{rate}/(4 * F_{rate}), max\{T_{i,j}, (2 \cdot T_{rate})/F_{rate}\}\} \tag{8}$$

5.3 Quantization Parameter Determination

5.3.1 Quantization Parameter for Inter-frames

To calculate the encoding parameter, Quantization Parameter (QP), for an inter-frame, we adopt the following pixel-wise unified rate-quantization (R-Q) model [26, 27]:

$$\frac{T_{i,j}}{N_{pixel}} = MAD_{i,j}^{P} \times \left(\frac{x_{i,j}^{1}}{\left(QP_{i,j}\right)^{2}} + \frac{x_{i,j}^{2}}{QP_{i,j}}\right) \tag{9}$$

where $QP_{i,j}$ denotes the current inter-frame's QP, N_{pixel} is the number of pixels in a frame, $MAD_{i,j}^{P}$ is the predicted Mean Absolute Difference for the current frame. $X_{i,j}^{1}$ and $X_{i,j}^{2}$ are model parameters of the quadratic model, which are updated after encoding each frame. $MAD_{i,j}^{P}$ is predicted based on the actual MAD value of its previous frame [26, 27]. Once $MAD_{i,j}^{p}$, $T_{i,j}$, $X_{i,j}^{1}$, and $X_{i,j}^{2}$ are known, $QP_{i,j}$ for the current frame can be calculated from Eq. (9).

5.3.2 QP Determination for Intra-frames

Intra-frames do not follow the same QP calculation as inter-frames. Same as H.265/HEVC RC, our bit allocation method does not allocate target bits to an intra-frame, nor calculate its QP based on the R-Q model. Intra-frames have their own form of QP calculation. Instead, a different approach for calculating the QP is proposed based on averaging the QPs from previous inter-coded frames with minor adjustments [46]. Though simple, this method is effective and quick to implement. The formula for this approach is described as:

$$QP_I = QP_{Inter}^{ave} + \Delta Q, \tag{10}$$

where QP_I denotes the Quantization Parameter of the current I-frame, and QP_{Inter}^{ave} is the average QP of previous m inter-coded frames, with m being set to 3 in our experiments. At the beginning, the initial value of ΔQ is 1.0. It is updated during the coding process by:

$$\Delta Q = \Delta Q + (PSNR_{I,k1} - PSNR_{inter,k1}^{ave})/\theta, \tag{11}$$

where k_1 is the coding time of the last I-frame, $PSNR_{I,k1}$ is the PSNR of the last I-frame, and $PSNR_{inter,k1}^{ave}$ is the average PSNR of 3 inter coded frames before the last I-frame. θ is a tuning parameter and is empirically set to 16 in our experiments.

In order to obtain balanced coding quality among intra-frames and inter-frames, we compare coding qualities between an intra-frame and its previous inter-frames. From Eq. (11), we can see that if $PSNR_{inter,k1}^{ave}$ is lower than $PSNR_{I,k1}$, we increase ΔQ and thus increase QP for the current I-frame so as to decrease its coding quality. However, if $PSNR_{inter,k1}^{ave}$ is higher than $PSNR_{I,k1}$, we need to decrease the QP of this I-frame to improve its coding quality.

After QP is determined, the encoder uses this QP to encode the current frame. The encoded bits will then be sent to the channel directly. Please note that, when buffer occupancy is very high, conventional rate control algorithms skip encoding frames to reduce buffer fullness and avoid buffer overflow. H.265/HEVC RC assumes unlimited buffer size, which is impractical, and thus never skips a frame. Our algorithm never uses a buffer and naturally omits buffer and frame skipping control, which can reduce encoding delay and improve real time response to a large degree.

5.3 Summary of the Proposed Algorithm

The summary of the algorithm is broken down into these steps:

Step 1. Allocate target bits to a GOP using Eq. (3)

Step 2. Allocate the initial target bits to an inter-frame using Eq. (4).

Step 3. Calculate the bit adjustment according to the PID controller by Eq. (2)

Step 4. Use the PID bit adjustment to modify the initial target bits for an inter-frame by Eq. (7)

Step 5. Use Eq. (8) to restrict the target bits for an inter-frame into a reasonable range.

Step 6. Compute a QP for an inter-frame using Eq. (9)

Step 7. Calculate a QP for an intra-frame using Eq. (10).

Step 8. Encode a frame using the obtained QP.

Step 9. After encoding, update the R-D model, adjust the weights, and compute the bit error between the target bit budget and actual coding bits, using Eq. (1). (5), (6), (11).

Step 10. Go to step 1 or 2 to encode the next GOP/frame until the end.

The flowchart of the proposed algorithm is illustrated in Figure 5.1 below.

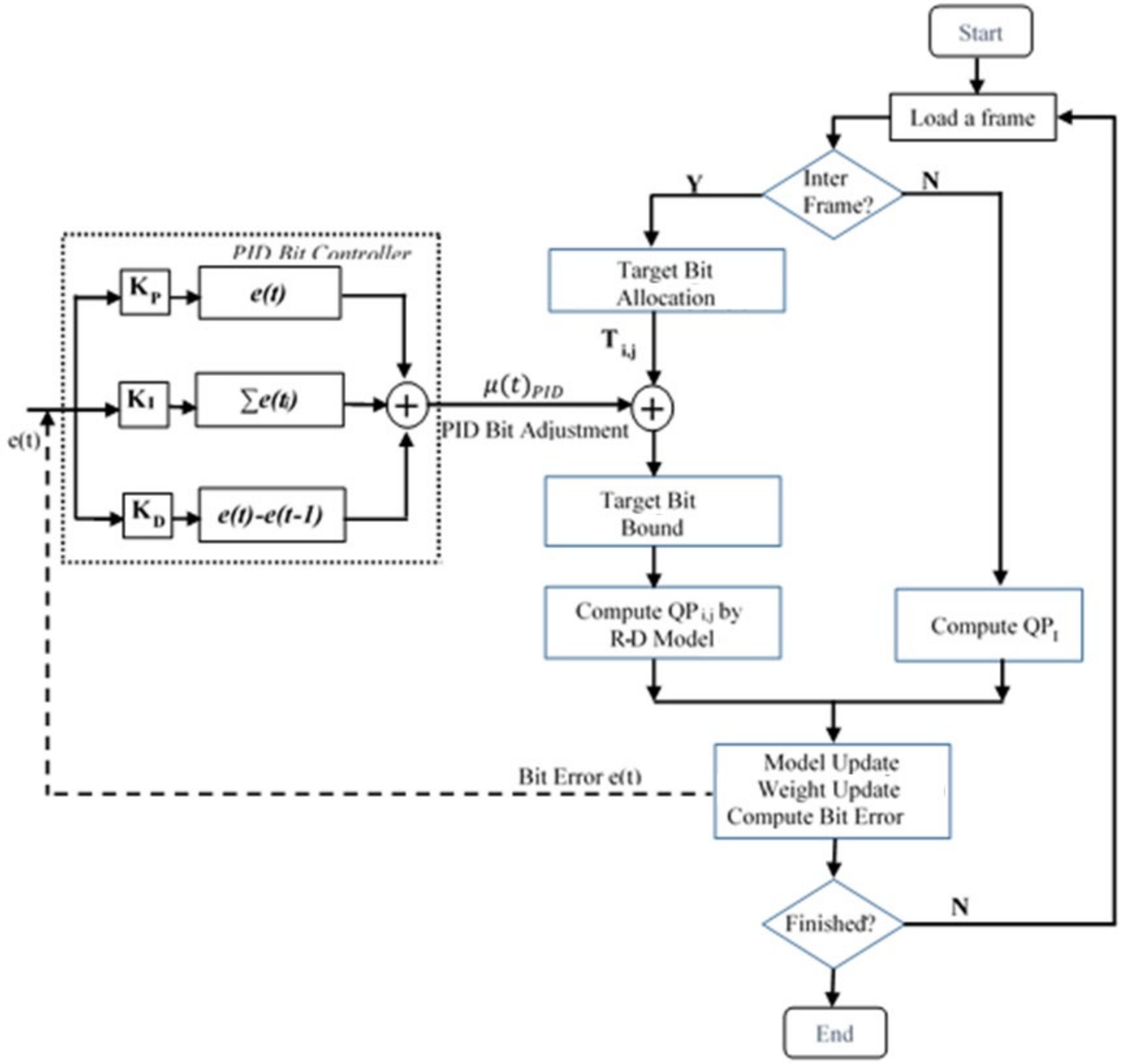


Figure 5.1: The Flowchart of the Proposed Rate Control Algorithm

CHAPTER 6 AUTOMATION TOOL FOR COLLECTING EXPERIMENTAL DATA

Collecting and formatting data is a very time consuming and tedious process. Spending time manually collecting and arranging data takes time away from making improvements to the algorithm itself. Furthermore, the time to encode a video can be long depending on the resolution and length of the video. This problem exacerbates itself when encoding multiple video test sequences with various target bit rates. Due to this, automating the testing process was a necessity. Because most of these tasks were not overly complex and could be repeated, in this research, I developed a suite of automation tools to speed up the process of data collection and formatting.

I used Perl when developing these tools due to it having very good excel related modules, built in regular expression support, and a quick implementation time. The environment used when running these tests was Strawberry Perl, which is a Perl environment that runs on Windows and includes several preinstalled modules. Using Strawberry Perl sped up the process of installing and running these tools on multiple computers.

6.1 Automation Overview and Implementation

There were three main objectives when I developed the automation tools:

1) Create a system in which configuring tests is quick and easy.
2) Design a system that could run many test sequences with various target encoding rates sequentially.

3) Format the resultant data into a table ready form and automatically generate data graphs.

These three objectives were the focus when designing the automation tools. The first objective, creating a system for configuring tests, was critical to the overall process since we planned to test many video sequences with a variety of configurations. For this purpose, a comma-separated value (CSV) file was used to contain metadata information on each test being performed. This information includes the resolution of the video, the frame rate, target encoding rate, the GOP structure, path to the raw video, etc. Making this CSV file serves two purposes: it makes the data easy to parse, and it is a supported file type in excel which allows us to quickly add/edit each test. Figure 6.1 shows the structure and contents of this file.

	A	B	C	D	E	F	G	H	I	J	K
1	File Name	YUV File	FPS	Frame to be Encoded	Width	Height	Intra Period	GOP Size	GOP Structure	QP	TargetBitrate
2	Class_C_Test1	C:\Keiba_832x480_30.yuv	30	200	832	480	60	4	C:\PBPB.txt	40	1024000
3	Class_C_Test2	C:\Keiba_832x480_30.yuv	30	200	832	480	60	4	C:\PBPB.txt	40	1536000
4	Class_C_Test3	C:\Keiba_832x480_30.yuv	30	200	832	480	60	4	C:\PBPB.txt	40	2048000
5	END										

Figure 6.1: Metadata configuration file contents

Row 1 in Figure 6.1 is ignored since it contains column names that are there for our convenience. Each row represents an individual test and contains the necessary information to run the H.265/HEVC encoder. Column A describes the resultant folder name where the test data will be stored. By default, the directory where these tests are stored is in a predetermined results folder that is set within the automation program itself. Column B contains the path to the raw video file to be encoded, while columns C – F contain information on the video to be encoded. Columns G – K contain information for configuring the encoder such as the intra period, GOP structure and size, initial QP value, and the target encoding bitrate. Adding new tests involves inserting a new row before the

END tag and adding the necessary configuration information. The *END* tag signifies the end of tests to run.

Since running tests through the H.265/HEVC encoder takes a substantial amount of time, automating the execution of tests allows it to run continuously and save time. This initial process is responsible for setting up the test, running the encoder, and saving the resultant test data. It reads in the metadata configuration CSV file that contains all the information needed to run the encoder. To run the encoder, two things are needed: a *main* configuration file, and a *per sequence* configuration file. The *main* configuration file contains various parameters related to the encoder itself such as the GOP structure, GOP size, and intra period. The *per sequence* configuration file relates directly to the raw video being encoded and contains metadata information specifically relating to the video such as the width, height, frame rate, etc.

The automation process first reads the metadata configuration file and skips the first row. Then, it will read the next available line in the file. It will read the information in each column and build the two configuration files necessary to run the encoder. As mentioned previously, the *main* configuration file contains information pertinent to encoder settings. Checking Figure 6.1, this includes the intra period, GOP size, path of the GOP structure, QP, and target bitrate. This information is then inserted into a main configuration file that is based on one of the standard configuration files provided by H.265/HEVC. Next, the *per sequence* file is generated containing the video specific information such as the location of the YUV file, width/height, number of frames to encode, and the frame rate per second (FPS). The encoder is then executed using both generated configuration files. While the encoder is running, it is generating data files that contain the target bitrate, actual encoding bitrate, PSNR values, and QP values.

Once the encoding is finished, a folder is created using the file name from column A of Figure 6.1 and the data files are copied into it along with the two configuration files that were used. A summary is then generated that contains coding details in a form that can be inserted into a table. This process is repeated until the value *END* is found. Figure 6.2 provides an overview of the automation process.

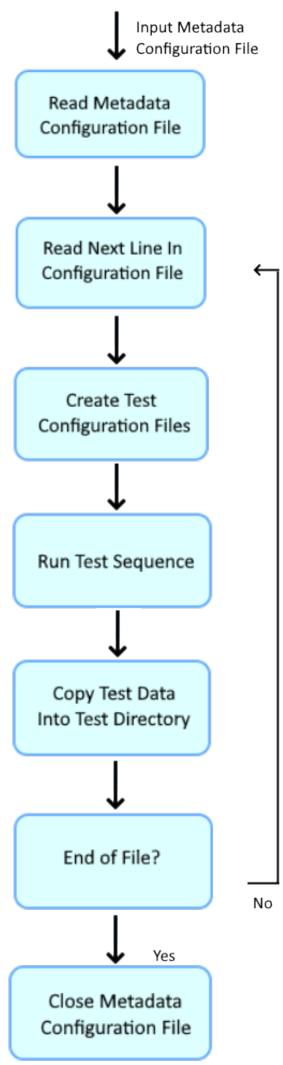


Figure 6.2: Automation Process Overview

6.2 Post Automation Data Formatting

Both the original H.265/HEVC encoder (HM8.0) and the revised encoder that contains our proposed rate control algorithm use this process with the same test configurations. Doing this allows us to directly compare the performance of the two rate control algorithms. This comparison is done through evaluating overall performance through PSNR averages and bitrate accuracy, along with showing these values over the course of video encoding. To speed up the process of formatting data, I also implemented another program which takes the results from the automation process and formats them into an excel workbook.

Once both the original HM8.0 encoder and the encoder with our proposed algorithm have finished the same set of tests, the resultant data folders are moved into their own respective directory. A new excel workbook is then created that will be used to store the formatted results. Next, the process will read each line of the metadata configuration file and create a new worksheet for each test listed. The data files for each test are then inserted into a table where the results for both encoders are listed side by side. Finally, two graphs are automatically generated: actual encoding bitrate over time and the PSNR values for each frame. Using these two graphs we can compare the quality and smoothness between the proposed RC algorithm and the one adopted by the original HM8.0 encoder. Figure 6.3 provides an overview of the data formatting process.

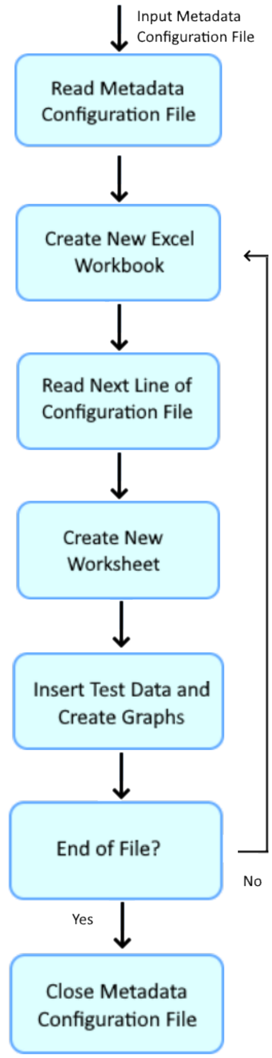


Figure 6.3: Data Formatting Overview

6.3 Data Collected

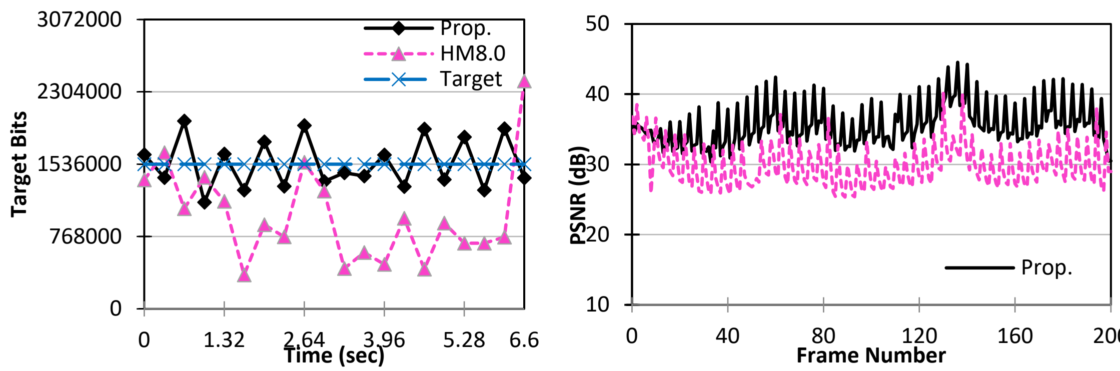
For evaluating the performance of the proposed algorithm, a variety of data points are collected and used for creating graphs and tables. I modified the H.265/HEVC encoder to output this data into separate data files that are collected by the automation process. Near the end of the encoding process, the data is combined into one file as shown by Table 6.1.

Table 6.1. Sample Results from Combined Data File

Frame Number	PSNR	Initial Target Bits	Final Target Bits	Encoded Bits	Alpha I	Alpha B	QP
0	36.7805	N/A	N/A	105352	3	0.5	32
1	34.4334	N/A	N/A	12016	3	0.5	34
2	35.3177	22755	N/A	24088	3	0.5	32
3	33.5819	68266	68029	70952	3	0.5	33

This data shows the performance over the course of encoding. It gives us insight into various components used in the algorithm such as our initial target bit estimation and the final target bit estimation. Alpha I and Alpha B refer to the weight parameters for I frames and B frames and are only updated after an I frame or B frame is encoded. PSNR represents that frames quality while QP is the quantization parameter used for that frame.

Once this data is obtained and inserted into an excel workbook, our program makes use of this data to automatically generate graphs. Figure 6.4 below is a sample of the graphs that are generated by our program.



(a) Keiba Bitrates (832x480, 1536k, IPP…IPP) (b) Keiba PSNR (832x480, 1536k, IPP…IPP)

Figure 6.4: Sample generated graphs

CHAPTER 7 EXPERIMENTS AND RESULTS

To conduct the performance evaluation, we implemented the proposed rate control algorithm on the HEVC reference software HM8.0 [49]. The performance of the proposed algorithm was compared with that of the HM8.0 RC algorithm adopted by H.265/HEVC [23, 24]. We conducted experiments under the Joint Collaborative Team on Video Coding (JCTVC) common test conditions and test sequences [50]. For fair evaluation, the comparisons between the two RC algorithms are carried out under the same encoding parameter configurations. We have tested standard sequences from different classes in different resolutions, ranging from 416x240 to 1280x720, with different frame rates: 30 frames per second (fps), 50fps, and 60fps. Each video had 200 frames encoded to allow for a more accurate comparison.

Three different sets of experiments were conducted using different temporal prediction structures along with different intra periods:

1) Intra period is 4 with repeating P frames (IPP...IPP), GOP size is 4; (IPPPIPPP...IPPP)

2) Intra period is 16 with 1 B-frame inserted between two reference frames (I- or P-frame), i.e.: IBPBP...IBPBP, GOP size is 8 (IBPBP...IBPBP)

3) Intra period is -1, only the 1st frame is used as an Intra-frame, and the rest of the frames are coded with P-frame, GOP size is 4 (IPPP…PP)

The standard measurements used for comparison was the overall rate control accuracy along with the Peak Signal-to-Noise Ratio (PSNR), in decibel units (dB), that measures the overall encoding quality. A higher PSNR value indicates a better coding quality, and vice versa.

The percentage error for RC, E_{rc}, is calculated by:

$$E_{rc} = \frac{|T_{rate} - C_{rate}|}{T_{rate}} \times 100\%$$

where C_{rate} represents the actual output encoding bitrates. A lower percentage value represents a smaller amount of error and a higher RC accuracy. The PSNR Gain is computed by:

$$Gain = PSNR_{Prop.} - PSNR_{HM8.0},$$

where $PSNR_{Prop}$ represents the proposed algorithm's PSNR while $PSNR_{HM8.0}$ indicates HM8.0 RC's PSNR. A positive *Gain* demonstrates the PSNR improvement obtained by the proposed algorithm. *Diff*, the difference between HM8 RC error (HM8 E_{rc}) and our algorithm's error (Prop. E_{rc}), is obtained by: $Diff = HM8.0\ E_{rc} - Prop.\ E_{rc}$. A positive *Diff* value indicates that HM8's RC error is higher than ours, demonstrating that our algorithm has improved the RC accuracy. A negative value means the RC error was increased by our algorithm, indicating decreased RC accuracy.

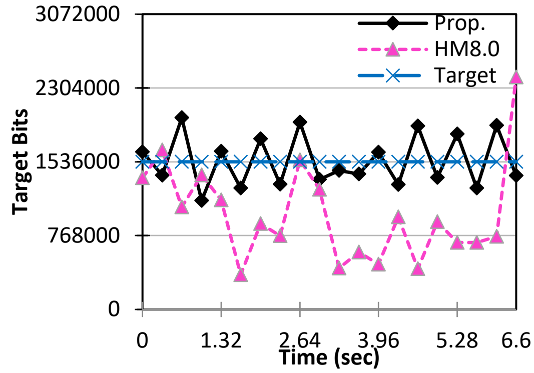
Table 7.1. Overall Performance Comparison – (Intra-Period=4, IPP…IPP)

Sequences	Target bitrate (Kbps)	Actual Encoding Bitrate (Kbps)		Rate Control Error (%)			PSNR (dB)		
		HM8.0	Prop.	HM8.0	Prop.	Diff.	HM8.0	Prop.	Gain
Keiba (832x480, 30 fps)	1024	837.75	1020.66	18.19	0.33	17.86	29.34	33.23	3.89
	1536	991.4	1545.31	35.46	0.61	34.85	29.83	35.66	5.82
Mobisode2 (832x480, 30 fps)	1536	1553.91	1550.31	1.17	0.93	0.24	41.09	42.07	0.98
	2048	763.83	2051.44	62.70	0.17	62.53	36.56	42.17	5.61
Flowervase (832x480, 30 fps)	1024	936.54	1027.95	8.54	0.39	8.15	34.54	35.47	0.93
	1536	1513.27	1518.52	1.48	1.14	0.34	36.07	36.19	0.12
BasketballDrill (832x480, 50 fps)	2048	1894.76	2042.01	7.48	0.29	7.19	28.99	32.84	3.85
	2560	2519.53	2574.08	1.58	0.55	1.03	29.98	34.53	4.55
PartyScene (832x480, 50 fps)	2048	2042.69	2050.45	0.26	0.12	0.14	23.28	25.61	2.32
	2560	2550.07	2547.29	0.39	0.50	-0.11	23.79	26.31	2.52
BasketballPass (416x240, 50 fps)	512	230.78	507.89	54.93	0.80	54.13	29.00	32.14	3.14
	1536	1390.26	1539.65	9.49	0.24	9.25	35.40	38.29	2.88
BlowingBubbles (416x240, 50 fps)	1024	476.86	1021.42	53.43	0.25	53.18	26.45	30.80	4.34
	2048	1686.91	2039.55	17.63	0.41	17.22	30.32	34.20	3.88
Keiba (416x240, 30 fps)	256	126.85	254.84	50.45	0.45	50.00	27.83	31.82	3.99
	512	327.29	515.13	36.08	0.61	35.47	31.74	35.52	3.78
Mobisode2 (416x240, 30 fps)	256	175.29	256.26	31.53	0.10	31.43	40.36	44.53	4.17
	1024	316.07	1007.85	69.13	1.58	67.55	42.82	49.33	6.50
FourPeople (1280x720, 60 fps)	2048	1202.06	2043.18	41.31	0.24	41.07	28.61	33.65	5.04
	2560	2160.04	2563.51	15.62	0.14	15.48	30.67	34.97	4.3
Johnny (1280x720, 60 fps)	2560	2340.41	2544.38	8.58	0.61	7.97	34.80	39.71	4.91
	3072	2855.92	3084.95	7.03	0.42	6.61	35.72	40.54	4.82
Average				24.2%	0.49%	23.71%	32.15	35.89	3.74

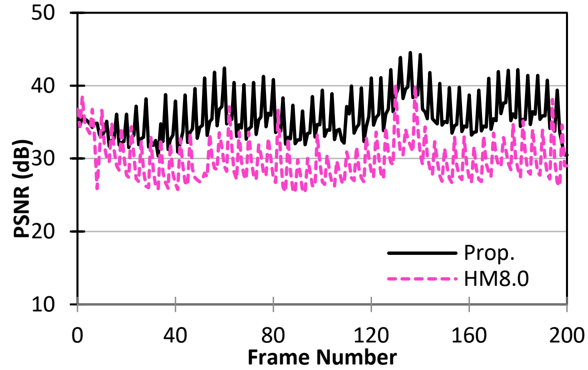
Table 7.1 presents the rate control performance for encoding structure 1 (IPPP...IPPP). From the results in Table 7.1, we can easily observe that our algorithm obtains more accurate rate control with usually higher average coding quality when compared with the HM8.0 RC. In some cases, HM8.0 RC algorithm is completely out of

control. For example, when the given target bitrate for the sequence Blowing Bubbles is 1024 Kbps, the actual encoding bitrate achieved by HM8.0 RC is 476.86 Kbps with an RC error of 53.43%, while our actual encoding bitrate is 1021.42 Kbps with only a very small error 0.25%. This shows that our algorithm's control ability is much stronger than that of HM8.0 RC. Overall, the average rate control errors of our algorithm and HM8.0 are 0.49% and 24.2% respectively, while the average coding quality gains of our algorithm over HM8.0 is 3.74 dB correspondingly.

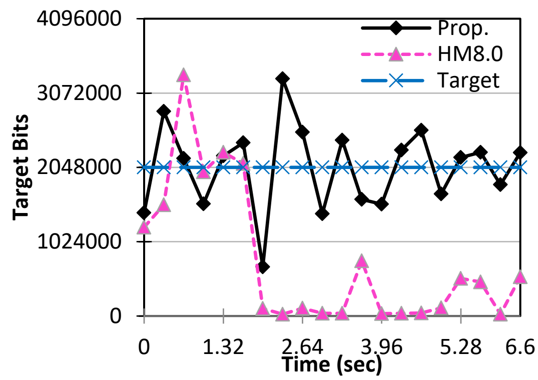
For the temporal prediction structure (IPP…IPP), Figure 7.1 presents the curves of the actual encoding bitrates and PSNRs. In these figures, the actual encoding bitrate obtained by the proposed aglorithm is indicated by the curve named Prop., while that achieved by HM8.0 RC is denoted by the curve of HM8.0. The target encoding bitrate is a constant, represented by a horizontal line named Target. From Figure 7.1 (a), (c), (e), (g), (i), (k), (m), (o), (q), (s), and (u), we can observe that our curves for actual encoding bitrate are closer to the target bitrates than the curves of HM8.0. This clearly demonstrates that our algorithm achieves better performance in rate control than HM8.0. Sometimes, for HM8.0 RC, the difference between the actual encoding bitrate and target bitrate is large, which indicates the weaker control ability of HM8.0. Figure 7.1 (b), (d), (f), (h), (j), (l), (n), (p), (r), (t), and (v) show that HM8.0's PSNR curves fluctuate more and are lower than our algorithm's PSNR curves. These demonstrate again that our algorithm can obtain better coding quality. All of these authenticate that the proposed RC algorithm outperforms HM8.0 RC.



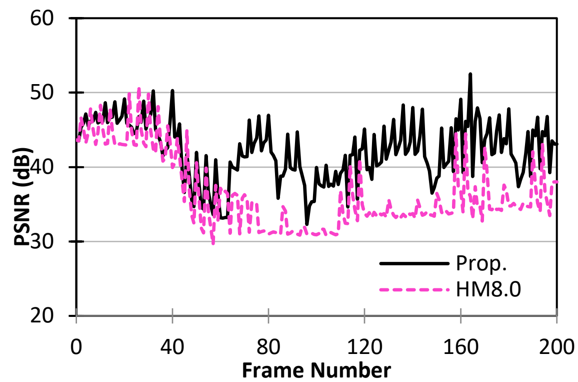
(a) Keiba Bitrates (832x480, 1536k)



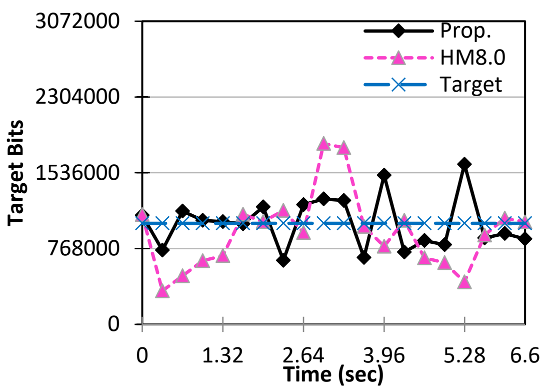
(b) Keiba PSNR (832x480, 1536k)



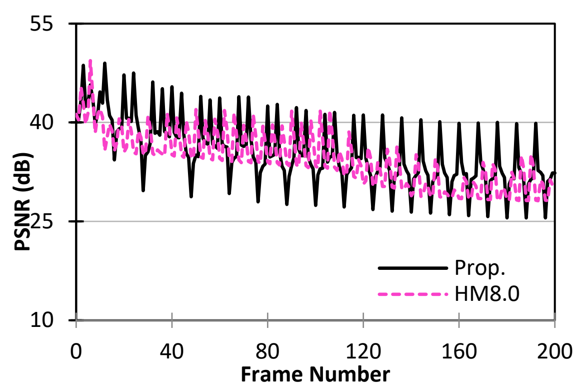
(c) Mobisode2 Bitrates (832x480, 2048k)



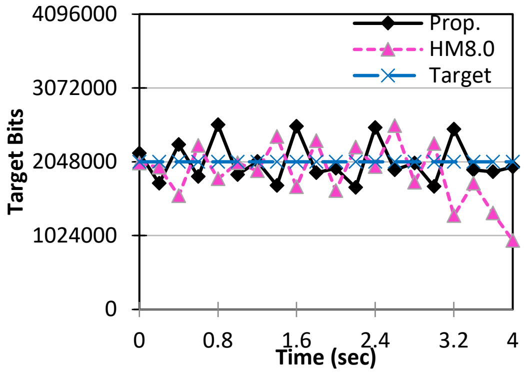
(d) Mobisode2 PSNR (832x480, 2048k)



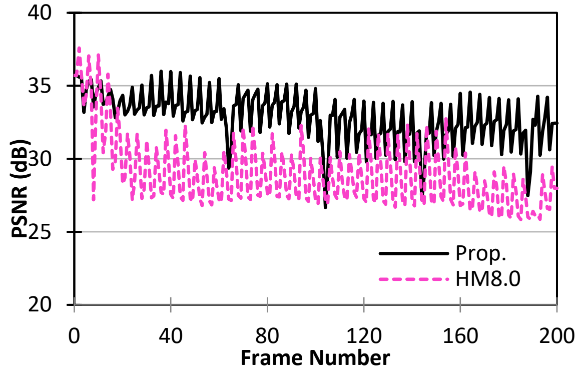
(e) Flowervase Bitrates (832x480, 1024k)



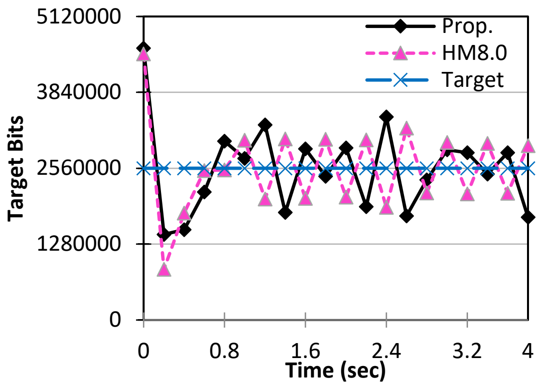
(f) Flower PSNR (832x480, 1024k)



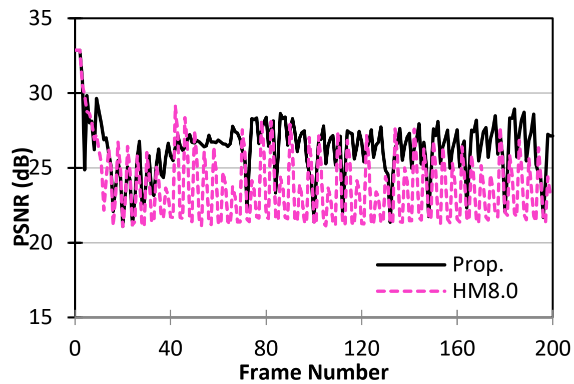
(g) BasketballDrill Bitrates (832x480, 2048k)



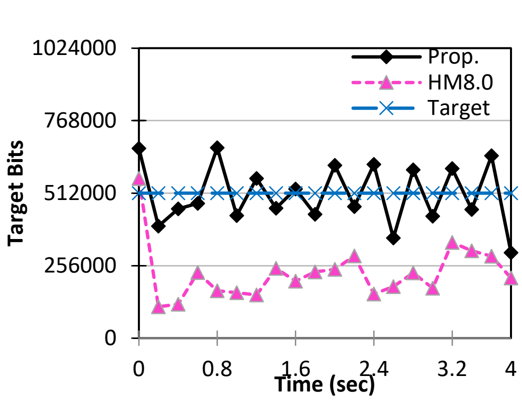
(h) BasketballDrill PSNR (832x480, 2048k)



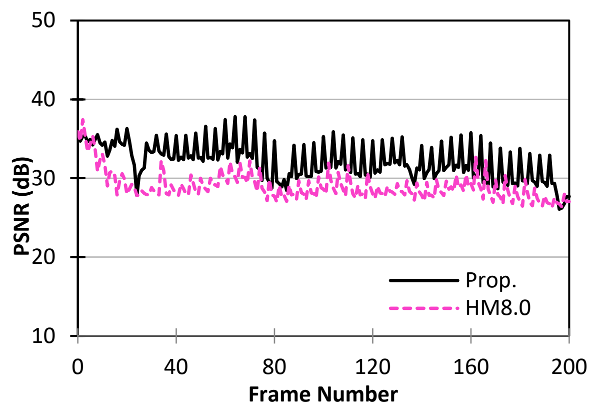
(i) PartyScene Bitrates (832x480, 2560k)



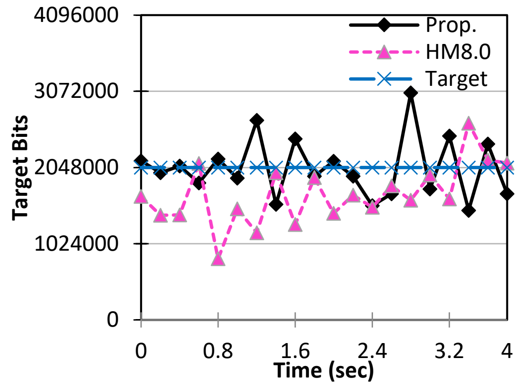
(j) PartyScene PSNR (832x480, 2560k)



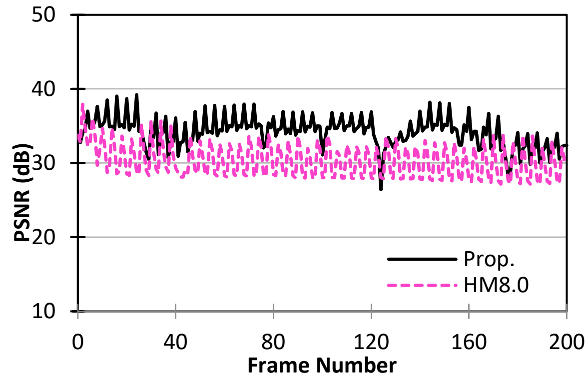
(k) BasketballPass Bitrates (416x240, 512k)



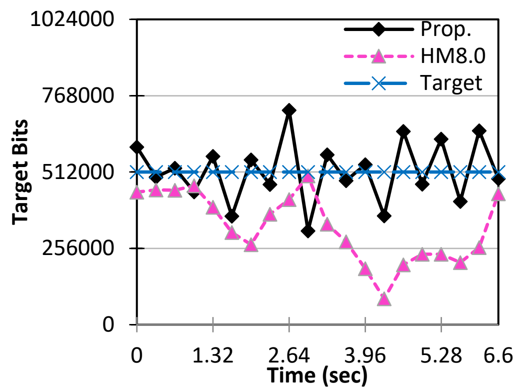
(l) BasketballPass PSNR (416x240, 512k)



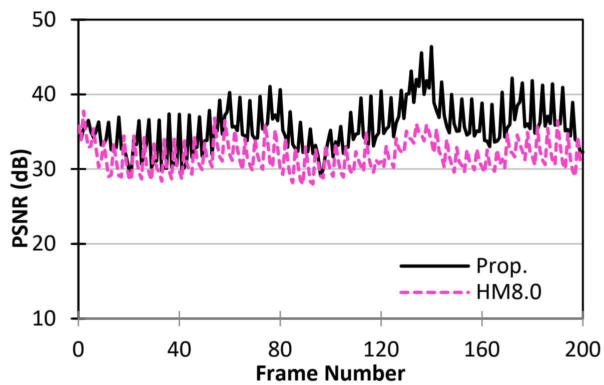
(m) BlowingBubbles Bitrates (416x240, 2048k)



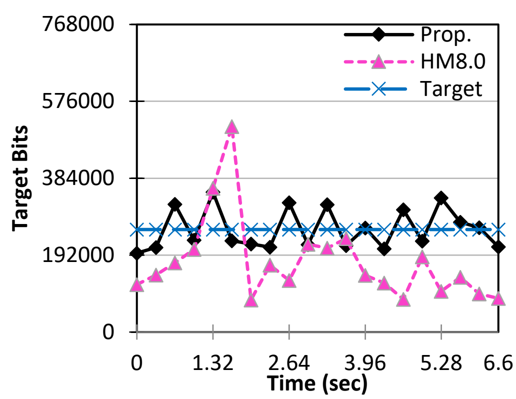
(n) BlowingBubbles PSNR (416x240, 2048k)



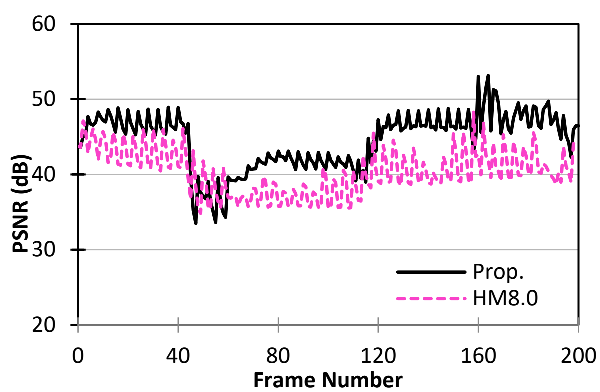
(o) Keiba Bitrates (416x240, 512k)



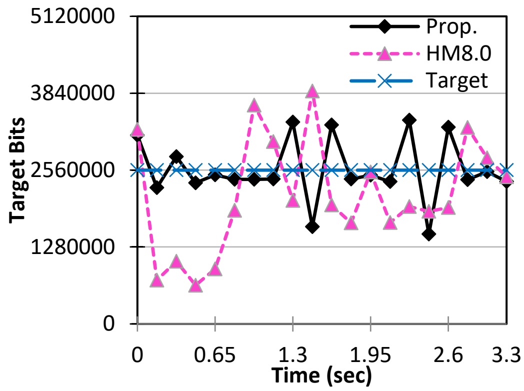
(p) Keiba PSNR (416x240, 512k)



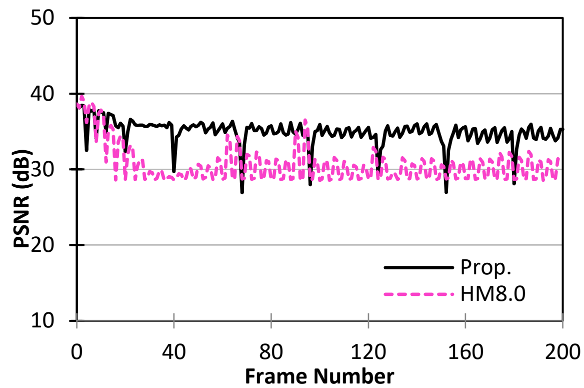
(q) Mobisode2 Bitrates (416x240, 256k)



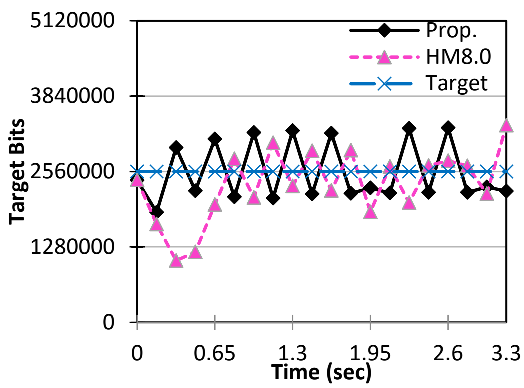
(r) Mobisode2 PSNR (416x240, 256k)



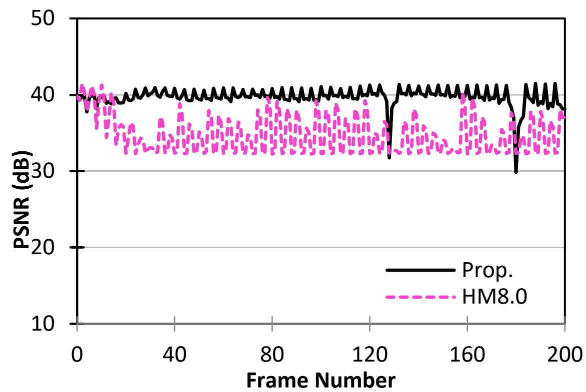
(s) FourPeople Bitrates (1280x720, 2560k)



(t) FourPeople PSNR (1280x720, 2560k)



(u) Johnny Bitrates (1280x720, 2560k)



(v) Johnny PSNR (1280x720, 2560k)

Figure 7.1: Experimental Results for Video Sequences Encoded at Various Bitrates (IntraPeriod=4, IPP...IPP)

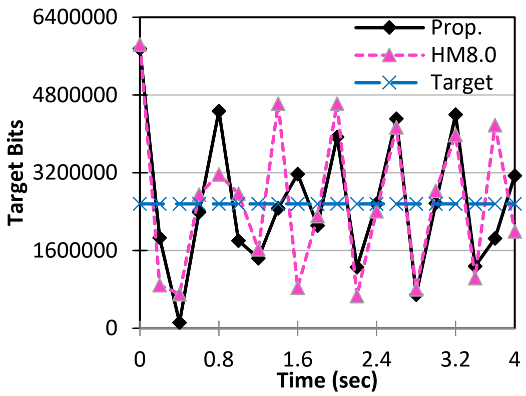
Table 7.2 shows the overall RC performance for temporal prediction structure 2 (Intra Period=16, IBP…IBP) and for temporal prediction structure 3 (IPP…PP). The details for the target bitrates, actual encoded bitrates, and frame PSNRs are illustrated in Figure 7.2. It is very complicated to perform rate control for the encoding structure 2 since all three frame types are involved. However, the results in Table 7.2 and Figure 7.2 demonstrate that our proposed algorithm is usually more effective at obtaining a higher average PSNR and more accurate rate control than HM8.0 RC. From Figure 7.2 (b), (d), (f), (h), (j), (l), (n), and (p), we can observe that in the PSNR curve of HM8.0 RC, there is greater fluctuation as its intra-coded frame's PSNR drop a lot when compared to the inter-coded frames. This indicates that HM8.0 RC is less efficient in handling intra-frames.

However, the PSNR curve of the proposed RC algorithm is much smoother, indicating our algorithm can handle intra-frames more efficiently.

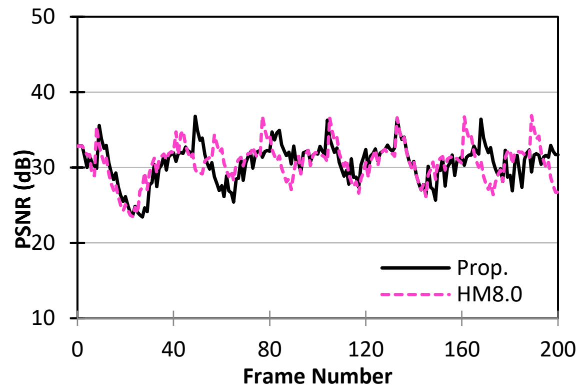
Table 7.2. Overall Performance Comparison – (Intra-Period=16, IBP) & (Intra-Period=-1, IPP)

Sequences	Target bitrate (Kbps)	Actual Encoding Bitrate (Kbps)		Rate Control Error (%)			PSNR (dB)		
		HM8.0	Prop.	HM8.0	Prop.	Diff.	HM8.0	Prop.	Gain
PartyScene (832x480, 50 fps, IPP…PP)	2560	2609.00	2580.61	1.91	0.81	1.10	30.72	30.60	-0.12
Flowervase (832x480, 30 fps, IPP…PP)	1536	1546.70	1532.02	0.70	0.26	0.44	41.77	41.81	0.04
PartyScene (832x480, 50 fps, IPP…PP)	3072	3151.62	3080.02	2.59	0.26	2.33	31.04	31.41	0.37
Mobisode2 (832x480, 30 fps, IPP…PP)	2048	2061.66	2051.13	0.67	0.15	0.52	45.03	45.10	0.07
BlowingBubbles (416x240, 50 fps, IPP…PP)	1536	1537.76	1537.03	0.12	0.07	0.05	36.56	36.62	0.06
Mobisode2 (416x240, 30 fps, IPP…PP)	1024	1031.95	1022.86	0.78	0.11	0.67	50.23	50.03	-0.20
Keiba (832x480, 30 fps, IBP…IBP)	1024	1015.76	1031.26	0.81	0.71	0.10	35.56	35.68	0.12
Keiba (832x480, 30 fps, IBP…IBP)	1536	1522.60	1540.99	0.87	0.33	0.54	36.91	37.38	0.47
Flowervase (832x480, 30 fps, IBP…IBP)	1024	1011.04	1025.62	1.27	0.16	1.11	39.35	40.27	0.92
Flowervase (832x480, 30 fps, IBP…IBP)	1536	1517.75	1534.28	1.19	0.11	1.08	40.15	41.25	1.10
BlowingBubbles (416x240, 50 fps, IBP…IBP)	1024	1015.20	1030.90	0.86	0.67	0.19	33.92	34.43	0.51
BlowingBubbles (416x240, 50 fps, IBP…IBP)	1536	1525.54	1536.30	0.68	0.02	0.66	35.55	35.87	0.32
BlowingBubbles (416x240, 50 fps, IBP…IBP)	2048	2031.95	2049.87	0.78	0.09	0.69	36.77	37.32	0.55
Johnny (1280x720, 60 fps, IBP…IBP)	2048	2015.26	2067.28	1.60	0.94	0.66	41.51	42.07	0.56
Johnny (1280x720, 60 fps, IBP…IBP)	2560	2519.96	2581.61	1.56	0.84	0.72	42.08	42.52	0.44
Johnny (1280x720, 60 fps, IBP…IBP)	3072	3035.13	3104.04	1.20	1.04	0.16	42.36	42.73	0.37
Average				1.10%	0.40%	0.70%	38.72	39.07	0.35

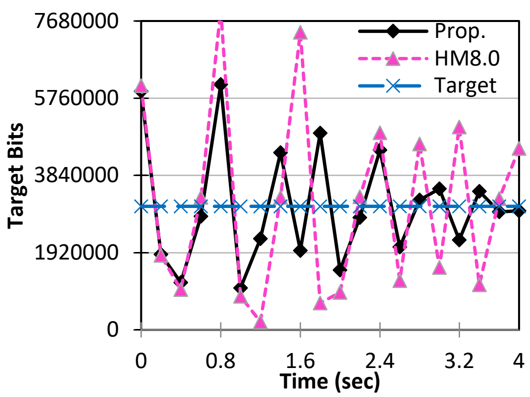
Please note that encoding structure 3 (IPP…PP) is the simplest case in RC since only P-frames need to be controlled. One can see from Table 7.2 and Figure 7.2 that, when compared with HM8.0 RC, the proposed algorithm has also achieved slightly better or similar performance on rate control accuracy and average PSNR.



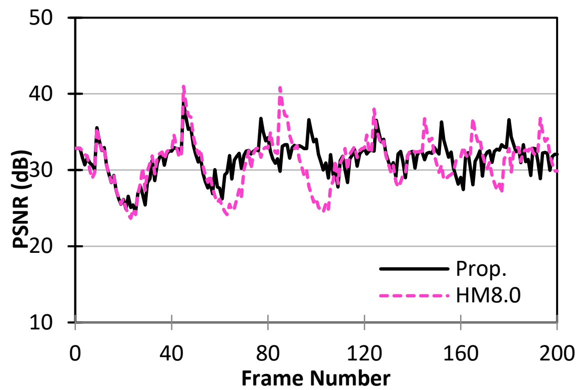
(a) PartyScene Bitrates (832x480, 2560k, IPPP)



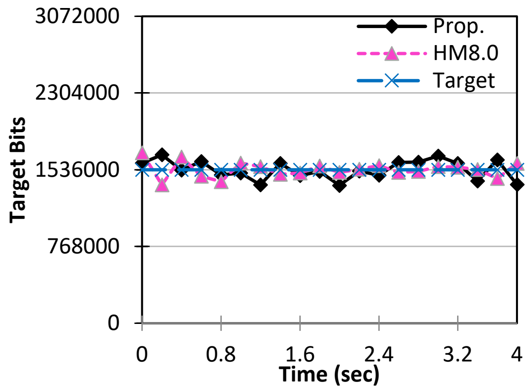
(b) PartyScene PSNR (832x480, 2560k, IPPP)



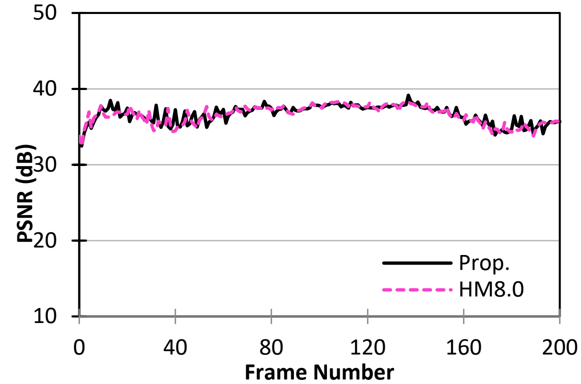
(c) PartyScene Bitrates (832x480, 3072k, IPPP)



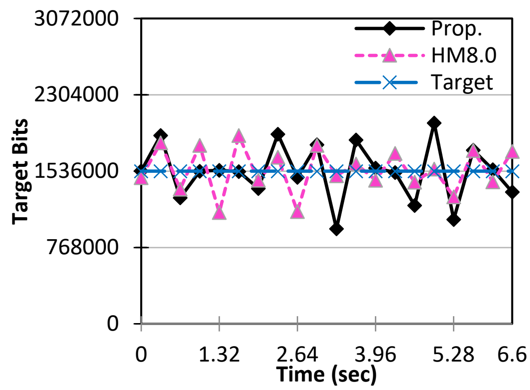
(d) PartyScene PSNR (832x480, 3072k, IPPP)



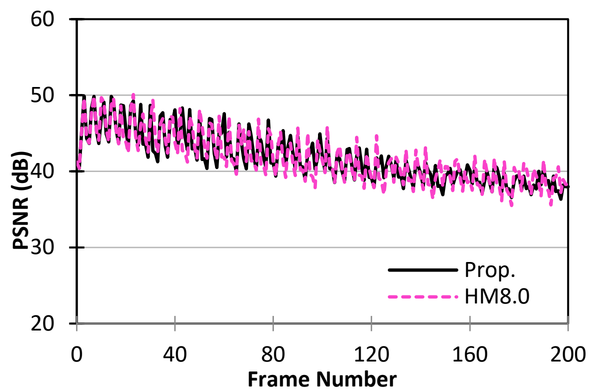
(e) BlowingBubbles Bitrates (416x240, 1536k, IPPP)



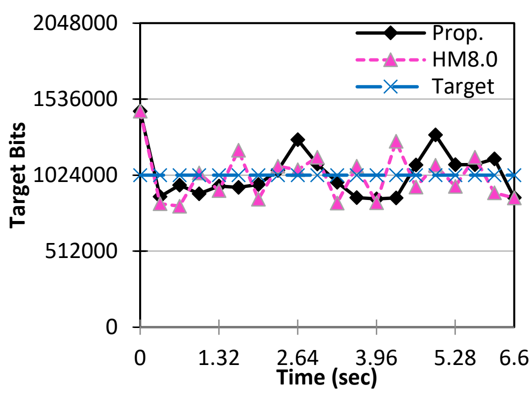
(f) BlowingBubbles PSNR (416x240, 1536k, IPPP)



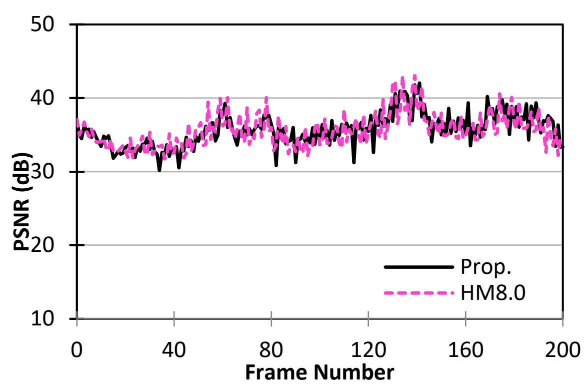
(g) Flowervase Bitrates (832x480, 1536k, IPPP)



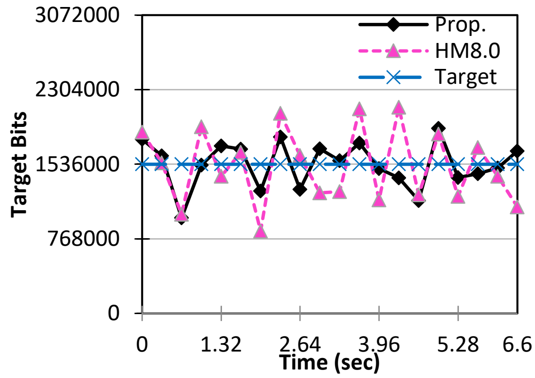
(h) Flower PSNR (832x480, 1536k, IPPP)



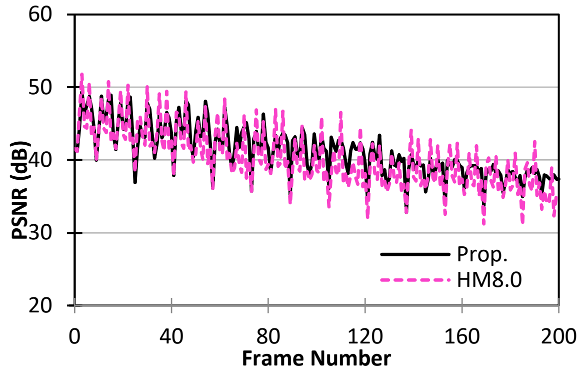
(i) Keiba Bitrates (832x480, 1024k, IBP)



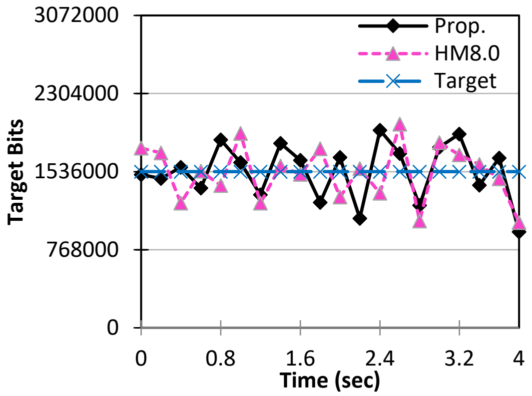
(j) Keiba PSNR (832x480, 1024k, IBP)



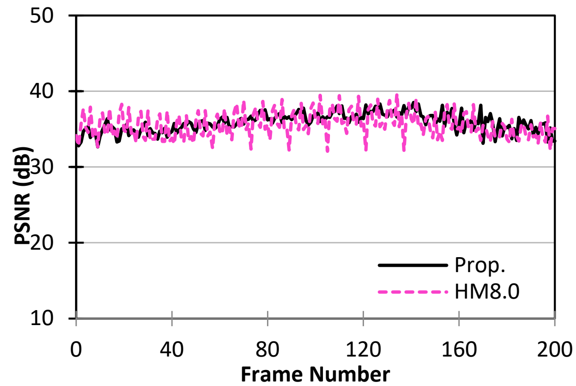
(k) Flowervase Bitrates (832x480, 1536k, IBP)



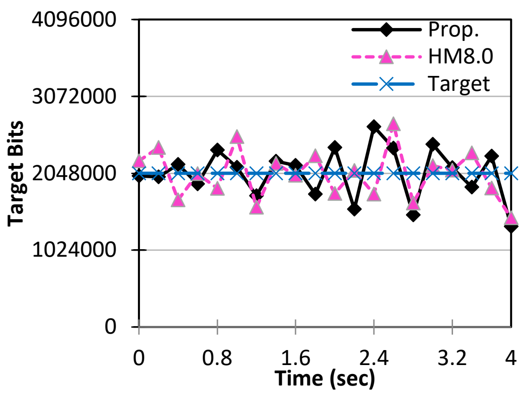
(l) Flower PSNR (832x480, 1536k, IBP)



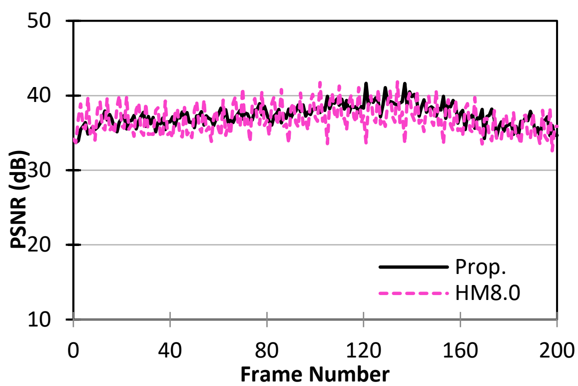
(m) BlowingBubbles Bitrates (416x240, 1536k, IBP)



(n) BlowingBubbles PSNR (416x240, 1536k, IBP)



(o) BlowingBubbles Bitrates (416x240, 2048k, IBP)



(p) BlowingBubbles PSNR (416x240, 2048k, IBP)

Figure 7.2: Experimental Results for Video Sequences Encoded at Various Bitrates (Intra-Period=16, IBP…IBP) & (Intra-Period= -1, IPP…PP)

CHAPTER 8 CONCLUSIONS

In video compression, rate control plays an essential role in regulating compressed bit streams to satisfy predefined constraints, such as network bandwidth, buffer size, and optimal visual quality. The primary research objective of this thesis was to investigate rate control techniques for the current video compression standard – H.265/HEVC. To conclude this thesis, our major contributions of this research are summarized first followed by the discussion of potential future work.

First, we studied the concepts and principles in video compression and rate control. Second, we investigated various video rate control algorithms, particularly for H.265/HEVC. Third, we analyzed the complex source codes of the H.265/HEVC encoder with a focus on rate control. Fourth, we studied the PID control theory and developed a direct PID-based bit control for inter-frames to reduce the deviation between target bits and actual coding bits. Fifth, we proposed a bufferless rate control algorithm for H.265/HEVC video compression. Sixth, we implemented an automation tool to collect encoding results and calculate rate control statistics automatically. Finally, we implemented our proposed methods and algorithms in the H.265/HEVC encoder and evaluated the performance.

This research proposed several novel ideas for rate control. First, it does not adopt a buffer in smoothing out compressed bit streams and adjusting target bits. Second, it applies a Proportional-Integral-Derivative approach to adjust the initial target bit budget and directly control the actual coding bitrate to achieve accurate rate control. The proposed algorithm has low complexity and low encoding delay, which is suitable for real-time video applications. The experimental results have shown that, when compared with the rate control algorithm adopted by H.265/HEVC, the proposed rate control algorithm achieves better performance in terms of rate control accuracy and encoding quality, resulting in an overall improvement to encoding.

Our future work will concentrate on adapting the proposed algorithm to work at different coding unit levels and extending it to the newest H.265/HEVC reference software. The algorithm can also be further optimized by tuning weight parameters that are associated with the PID controller. Giving more or less weight to the different components of the PID controller can have a significant effect on the end coding results. Finally, the other structures that the algorithm relies on, such as target bit estimation, can be changed based on new methodologies and approaches.

BIBLIOGRAPHY

[1] G. J. Sullivan, J.-R. Ohm, W. Han and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[2] Z.-N. Li and M. S. Drew, Fundamentals of Multimedia, Prentice Hall, 2004.

[3] B. Furht, J. Greenberg and R. Westwater, "Motion Compensation and Estimation," in *Motion Estimation Algorithms for Video Compression*, New York, Springer Science+Business Media, 1997, pp. 49-82.

[4] K. Sayood, Introduction to Data Compression, Burlington, MA: Morgan Kaufmann, 2017.

[5] ITU-T Recommendation H.261: "Video Codec for Audiovisual Services at px64 kbit/s", Geneva, 1990., Geneva, 1990.

[6] ITU-T Recommendation H.263, Version2: “VideoCoding for Low Bitrate Communication”, Geneve 1998.

[7] ISO/IEC JTC1/SC29, “Coding of Moving Pictures and Associated Audio for Digital Storage Media up to about 1.5Mbits/s,” ISO/IEC 11172-2, International Standard, Nov. 1992.

[8] MPEG-2 Test Model 5, Doc. ISO/IEC JTCI/SC29/WG11/93-400, Apr. 1993.

[9] MPEG-4 Video Verification Model V8.0, ISO/IEC JTC1/SC29/WG11 N3093, "Coding of Moving Pictures and Audio", Maui, Dec. 1999.

[10] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 13, no. 7, pp. 560-576, Jul. 2003.

[11] M. T. Sun and A. R. Reibman, Compressed Video over Networks, New York, NY: Marcel Dekker, Inc., 2001.

[12] S. Xu, M. Yu, S. Fang, Z. Peng and Z. Wang, "New Rate Control Optimization Algorithm for HEVC Aiming at Discontinuous Scene," *WSEAS Transactions on Computers,* vol. 14, pp. 598-606, 2015.

[13] Z. Chen and K. Ngan, "Recent advances in rate control for video coding," *Signal Processing: Image Communication,* vol. 22, pp. 19-38, 2006.

[14] A. Ortega and K. Ramchandran, "Rate-Distortion Methods for Image and Video Compression," *IEEE Signal Processing Magazine,* vol. 15, no. 6, pp. 23-50, 1998.

[15] T. Berger, "Rate Distortion Theory and Data Compression," in *Advances in Source Coding*, New York, NY, Springer, 1975, pp. 1-39.

[16] T. Kim, B. Roh and J. Kim, "An accurate bit-rate control for real-time MPEG video encoder," *Signal Processing: Image Communication,* vol. 15, no. 6, pp. 479-492, Mar. 2000.

[17] S. Vetrivel, K. Suba and G. Athisa, "An Overview of H.26x Series and its Applications," *International Journal of Engineering Science and Technology,* vol. 2, no. 9, pp. 4622-4631, Sep. 2010.

[18] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg and D. J. LeGall, "MPEG-2 Overview," in *MPEG Video Compression Standard*, New York, Springer, 1996, pp. 171-186.

[19] P. N. Tudor, "MPEG-2 video compression," *Electronics & Communication Engineering Journal,* vol. 7, no. 6, pp. 257-264, Dec. 1995.

[20] S. Aramvith and M. Sun, "MPEG-1 and MPEG-2 Video Standards," in *Handbook of Image and Video Processing*, San Diego, CA, Academic Press, 2000, pp. 597-611.

[21] Z. Wu, S. Xie, K. Zhang and R. Wu, "Rate Control in Video Coding," in *Recent Advances on Video Coding*, Rijeka, Croatia, InTech, 2011, pp. 79-116.

[22] J. R. Corbera and S. Lei, "Rate Control in DCT Video Coding for Low-Delay Communications," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 9, no. 1, pp. 172-185, Feb. 1999.

[23] H.-J. Lee, T. Chiang and Y.-Q. Zhang, "Scalable rate control for MPEG-4 video," *IEEE Transactions on Circuits Systems Video Technology,* vol. 10, no. 6, pp. 878-894, Sep. 2000.

[24] S. Ma, W. Gao, D. Zhao and Y. Lu, "A Study on the Quantization Scheme in H.264/AVC and Its Application to Rate Control," in *Pacific-Rim Conference on Multimedia*, Tokyo, Japan, 2004.

[25] Z. G. Li, F. Pan, K. P. Lim and G. N. Feng, "Adaptive Basic Unit Layer Rate Control for JVT," in *JVT-G012, 7th meeting*, Pattaya, Thailand, Mar. 2003.

[26] H. Choi, J. Yoo, J. Nam, D. Sim and I. Bajic, "Pixel-Wise Unified Rate-Quantization Model for Multi-Level Rate Control," *IEEE Journal of Selected Topics in Signal Processing,* vol. 7, no. 6, pp. 1112-1123, Jul. 2013.

[27] H. Choi, J. Nam, J. Yoo, D. Sim and I. V. Bajiæ, "Rate Control Based on Unified RQ Model for HEVC," in *8-th JCTVC Meeting*, San Jose, CA, USA, Feb. 2012.

[28] B. Li, H. Li, L. Li and J. Zhang, "Rate Control by R-Lambda Model for HEVC," in *11th Meeting JCTVC Meeting*, Shanghai, China, Oct 2012.

[29] B. Li, H. Li, L. Li and J. Zhang, "λ Domain Rate Control Algorithm for High Efficiency Video Coding," *IEEE Transactions on Image Processing,* vol. 23, no. 9, pp. 3841-3854, Jul. 2014.

[30] M. Wang, K. N. Ngan and H. Li, "An Efficient Frame-Content Based Intra Frame Rate Control for High Efficiency Video Coding," *IEEE Signal Processing Letters,* vol. 22, no. 7, pp. 896-900, Jul. 2015.

[31] S. Li, M. Xu, X. Deng and Z. Wang, "Weight-based R- λ Rate Control for Perceptual HEVC Coding on Conversational Videos," *Signal Processing Image Communication,* vol. 38, pp. 127-140, Oct. 2015.

[32] M. Wang, K. N. Ngan and H. Li, "Low-Delay Rate Control for Consistent Quality using Distortion-based Lagrange Multiplier," *IEEE Transactions on Image Processing,* vol. 25, no. 7, p. 2943–2955, Apr. 2016.

[33] W. Gao, S. Kwong, Y. Zhou and H. Yuan, "SSIM-based game theory approach for rate-distortion optimized intra frame ctu-level bit allocation," *IEEE Transactions on Multimedia,* vol. 18, no. 6, pp. 988-999, Mar. 2016.

[34] W. Gao, S. Kwong, H. Yuan and X. Wang, "DCT coefficient distribution modeling and quality dependency analysis based frame-level bit allocation for HEVC," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 26, no. 1, pp. 139-153, Jun. 2015.

[35] M. Zhou, X. Wei, S. Wang, S. Kwong, C.-K. Fong, P. Wong, W. Yuen and W. Gao, "SSIM-Based Global Optimization for CTU-Level Rate Control in HEVC," *IEEE Transactions on Multimedia,* vol. 21, no. 8, pp. 1921-1933, Jan. 2019.

[36] S. Li, M. Xu, Z. Wang and X. Sun, "Optimal Bit Allocation for CTU Level Rate Control in HEVC," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 27, no. 11, pp. 2409-2424, Nov. 2017.

[37] M. Zhou, Y. Zhang, B. Li and X. Lin, "Complexity Correlation-Based CTU-Level Rate Control with Direction Selection for HEVC," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM),* vol. 13, no. 4, Aug. 2017.

[38] Z. Liu, L. Wang, X. Li and X. Ji, "Optimize x265 Rate Control: An Exploration of Lookahead in Frame Bit Allocation and Slice Type Decision," *IEEE Transactions on Image Processing,* vol. 28, no. 5, pp. 2558-2573, May 2019.

[39] W. Gao, S. Kwong, Q. Jiang, C. Fong, P. H. W. Wong and W. Y. F. Yuen, "Data-Driven Rate Control for Rate-Distortion Optimization in HEVC Based on Simplified Effective Initial QP Learning," *IEEE Transactions on Broadcasting,* vol. 65, no. 1, pp. 94-108, Mar. 2019.

[40] W. Gao, S. Kwong and Y. Jia, "Joint machine learning and game theory for rate control in high efficiency video coding," *IEEE Transactions on Image Processing,* vol. 26, no. 12, pp. 6074-6089, Dec. 2017.

[41] K. H. Ang, G. Chong and Y. Li, "PID control system analysis, design, and technology," *IEEE Transactions on Control Systems Technology,* vol. 13, no. 4, pp. 559-576, Jun. 2005.

[42] Y. Sun and I. Ahmad, "A Robust and Adaptive Rate Control Algorithm for Objects based Video," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 14, no. 10, pp. 1167-1182, Oct. 2004.

[43] Y. Sun and I. Ahmad, "Asynchronous Rate Control for Multi-Object Videos," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 15, no. 8, pp. 1007-1018, Aug. 2005.

[44] Y. Sun, I. Ahmad, D. Li and Y.-Q. Zhang, "Region-Based Rate Control and Bit Allocation for Wireless Video Transmission," *IEEE Transactions on Multimedia,* vol. 8, no. 1, pp. 1-10, Feb. 2006.

[45] Y. Zhou, Y. Sun, Z. Feng and S. Sun, "PID-Based Bit Allocation Strategy for H.264/AVC Rate Control," *IEEE Transaction on Circuits and Systems II,* vol. 58, no. 3, pp. 184-188, Mar. 2011.

[46] Y. Sun, Z. Feng and R. R. Ginnavaram, "A Direct Non-Buffer Rate Control Algorithm for Real Time Video Compression," *International Journal of Multimedia Tools and Applications,* vol. 74, no. 17, pp. 6623-6639, Sep. 2015.

[47] S. Bennett, "A brief history of automatic control," *IEEE Control Systems Magazine,* vol. 16, no. 3, pp. 17-25, Jun. 1996.

[48] K. J. Åstrom and T. Hagglund, PID Controllers: Theory, Design, and Tuning, Research Triangle Park, NC: Instrument Society of America, 1995.

[49] I. K. Kim, K. McCann, K. Sugimoto, B. Bross and W. J. Han, "High Efficiency Video Coding (HEVC) Test Model 8 (HM 8) Encoder Description," in *Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 and ISI/IEC JTC 1/SC 29/WG 11*, Stockholm, Sweden, 2012.

[50] F. Bossen, "Common test conditions and software reference configurations," in *JCT-VC of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11*, San Jose, USA, Feb. 2012.