Re-Engineering Rate Distortion Optimisation in Modern Video Codecs Using a Per Clip Approach

Daniel Joseph Ringis

Supervisor: Prof. Anil Kokaram

Dr. François Pitié

Department of Electronic and Electrical Engineering

Trinity College Dublin

This dissertation is submitted for the degree of

Doctor of Philosophy

January 2023

I would like to dedicate this thesis to my supportive family.

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work. I agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement. I consent to the examiner retaining a copy of the thesis beyond the examining period, should they so wish (EU GDPR May 2018).

Daniel Joseph Ringis

January 2023

Acknowledgements

To borrow a line from one of my supervisor's thesis, "If you are reading this page, you are probably looking for your name, and you most likely deserve it, so thank you very much!"

I wish to express profound gratitude towards my supervisors Prof. Anil Kokaram and Dr. François Pitié who have somehow survived this process with me during a global pandemic and many a frantic email or meeting. You both have been immensely supportive and patient beyond what could be reasonably expected throughout this process and for that I thank you again and again. I would like to acknowledge the Ussher's studentship, Google Faculty Award and the Sigmedia Bursary who enabled this work to be done through funding.

I would like to thank the members of the Sigmedia group, in particular, my fellow researchers over the years spent toiling in Stack B or AAP; (there have been a lot) Sébastian Le Maguer, Matthew Roddy, Hugh O'Dwyer, Luke Ferguson, Marco Forte, George Sterpu, Ayushi Pandey, Mark Anderson, Ed Storey, Darren Ramsook, Varoun Hanooman, Vibhoothi, Clement Bled, Meegan Gower, Sam O'Conner-Russell, Ciaran Donegan, Sophia Rosney, and Xin Shu. Similarly, close friends, Simone Tomwing, Annabella Frank, Trisha Sirjoo, Sarah Hampton and Annie Humphrey, each of whom have been pivotal in encouraging me to complete this, even while an ocean away.

I would like to thank the various groups who have provided me with distractions over the years, BMT Fantasy, Dublin Dodgeball, Trinity Museum Players, Fanatix Basketball, Dublin Homers, Trinity Walton Club and the PGWA. Each group has provided me with much needed outlets for my mental and physical health over the years.

Most importantly, I would like to thank my family, Nicole, David and Sarah Henry, Lois Brian, Marissa, Isaiah, Ryan and Christina Ringis, and Scot De Silva, for their love and support, throughout this entire process. I could not have done this without you.

Abstract

The majority of internet traffic is video content. This drives the demand for video compression to deliver high quality video at low target bitrates. Optimising the parameters of a video codec for a specific video clip (per-clip optimisation) has been shown to yield significant bitrate savings.

One of the key challenges to be solved in the design of a practical codec is the trade-off between rate and distortion. A Lagrangian multiplier has been the adopted view in video codecs. In this approach the encoder makes a number of decisions in order to minimise a cost J, where $J = D + \lambda R$, commonly referred to as the rate distortion equation. As can be seen J combines both a distortion D (for a frame or macroblock) and a rate R (the number of coded bits for that unit) through the action of the Lagrangian multiplier λ. Different choices for λ result in different R/D trade-offs.

This research investigates the impact of adjusting the rate distortion equation on compression performance. A constant of proportionality, k, is used to modify the Lagrangian multiplier used in H.265 (HEVC) and VP9. Direct optimisation methods are deployed to maximise BD-Rate improvement for a particular clip. By doing this, we have shown that per-clip optimisation of the Lagrangian multiplier leads to BD-Rate improvement of up to 25% in a single clip using Direct Optimisation. The average BD-Rate improvement was approximately 2% across the 10k clip corpus. This direct optimisation comes at a high computational cost, but can be used as an estimate of the best possible improvement possible for a given video clip.

Because of this computationally expensive method which requires multiple measurement of rate distortion curves which meant in excess of fifty video encodes were used to generate that level of savings. This research then focuses on reducing the computational cost of repeated video encodes by using proxy systems or by reducing this computational load through prediction. Approximately 60% of the BD-Rate improvements found by direct optimisation using proxies or prediction were achieved using the reduced complexity systems. This effectively reduced the computational complexity ten fold. Overall, our system achieves BD-Rate improvement in approximately 90% of a large corpus with comparable results to the direct optimisation methods.

Table of contents

List of Acronyms

ABR - Adaptive Bitrate Streaming

AOM - Alliance for Open Media

B Frame - Bi-directional Predicted Frame

BD - Bjontegaard Delta

BD-Distortion - Bjontegaard Delta Distortion

BD-Quality - Bjontegaard Delta Quality

BD-Rate - Bjontegaard Delta Rate

CTU - Coding Tree Unit

CBR - Constant Bitrate

CRF - Constant Rate Factor

CNN - Convolutional Neural Network

CDF - Cumulative Distribution Function

DNN - Deep Neural Network

DFD - Direct Frame Difference

DCT - Discrete Cosine Transform

DASH - Dynamic Adaptive Streaming over HTTP

GOP - Group of Pictures

HD - High Definition

HEVC - High Efficiency Video Coding

I Frame - Intra-frame

LUT - Look Up Table

ML - Machine Learning

MB - Macroblock

MSE - Mean Squared Error

MPEG - Motion Pictures Expert Group

PSNR - Peak Signal to Noise Ratio

P Frame - Predicted Frame

RD - Rate - Distortion

RDO - Rate Distortion Optimisation

RQ - Rate - Quality

SSIM - Structural Similarity Image Matrix

UGC - User Generated Content

VBR - Variable Bitrate

VMAF - Visual Multimodal Assessment Fusion

Chapter 1

Introduction

The amount of user-generated video content has increased significantly (Wang et al., 2019) to the point that it represents almost 80% of all internet traffic (Cass, 2014). Internet video has become so prevalent that it has been segmented into multiple subcategories such as messaging, social media and video on demand (Sandvine, 2021). There is an increasing need for video to be delivered at high quality and low bitrate. This is not a new problem. Since the origins of broadcast digital video, the principal goal has been to deliver the best quality video at a target bitrate or to achieve the lowest possible bitrate for a target quality. The COVID-19 pandemic has further increased this need with an uptick of work-from-home driving video conferencing use cases (Feldmann et al., 2021). Video compression algorithms attempt to achieve this goal of delivering high-quality video at a low bitrate.

The Moving Pictures Expert Group (MPEG) is a group which is responsible for the original standardisation efforts of digital video compression since 1988 (Fogg et al., 2007). Computers and consumer electronics were beginning to expand into using digital video, as opposed to just text and graphics, compact discs were being used for digital storage, the teleconferencing industry was starting to mature and a transition from analog broadcast television to digital video was beginning. All of these things put together naturally led to a standardisation effort (Fogg et al., 2007). This effort led to the development of the

MPEG-1 (1991), MPEG-2 (1995) and MPEG-4 (1999) standards (ISO/IEC 11172, ISO/IEC 13818, ISO/IEC 14496)

While it is essential to note the history and context of past video standards, this work will be focused on internet video. The H.264 standard (ISO/IEC 14496-10), describes a method of coding video which provided better performance than any preceding standards and allowed for compressed video clips to take up less transmission bandwidth. It is also referred to as MPEG-4 Part 10 or Advanced Video Coding (AVC) (Richardson, 2011). This was frequently used in video streaming in the late 2000s and remains one of the most widely adopted standards for internet video as recently as 2016, as evidenced in the YouTube upload statistics (Kokaram et al., 2016; Laude et al., 2019).

The terminology used to describe video compression systems involves a distinction between implementations and technical standards. The term *codec* frequently refers to the implementation of both the *encoder* and *decoder* described in a video compression standard. The term *encoder* refers to the implementation of the tools used to compress raw video into a coded bitstream. The corresponding *decoder* is the implementation which allows that coded bitstream to be decoded into a viewable format for display.

Subsequent improvements to H.264, continued to focus on digital internet video and how to provide the best quality video delivered at the lowest possible rate. These standards are H.265, often referred to as High Efficiency Video Coding (HEVC) (ISO/IEC 23008) (Sullivan et al., 2012) and H.266, often referred to as Versatile Video Coding (VVC) (ISO/IEC 23090-3) (Karwowski et al., 2016). In 2013, Google, introduced royalty-free video codecs, VP9 (Mukherjee et al., 2013), and the Alliance for Open Media introduced AV1 (Chen et al., 2018), as open-source competitors to the licensed video compression standards created by MPEG (Richardson, 2017).

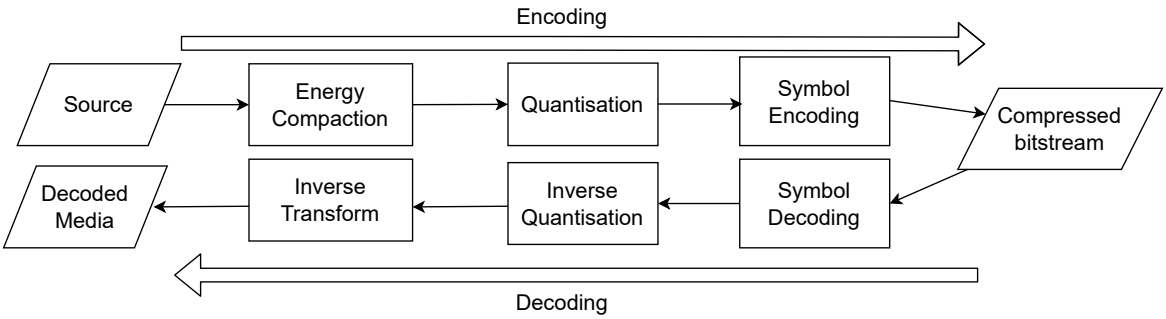

Fig. 1.1 *Source media goes through three steps in the encoding process to create a compressed bitstream. The energy compaction step, e.g. discrete cosine transform, quantisation and the symbol compaction step, e.g. Huffman encoding. The inverse of these steps are performed in decoding a compressed bitstream.*

1.1 Anatomy of Video Codecs

A video compression standard defines the syntax of the compressed bitstream and the functions to interpret and act upon reading these bits, but not the implementation of the encoding sub-modules used (Richardson, 2002). Typically this is expressed with the statement *"the decoding operation is standard but the encoding is not"*. Importantly, this framework enables competition and improvements in the codec market space. Although there is a common overall structure, each video compression standard has been defined for different use case scenarios. Modern video codecs (e.g. AV1 and H.266/VVC) have become quite complex (Corrêa et al., 2021). However, there is a general structure which holds true for these codecs. This structure is illustrated in Figure 1.1. This compression model is not limited to applications in media processing, but is used in other signal processing applications.

Starting with an uncompressed video, the energy compaction step is usually implemented as transform coding and is typically a Discrete Cosine Transform or a Discrete Wavelet transform. The main goal of this module is to transform your source, to as small as possible number of significant values (Richardson, 2002). The coefficients generated by the transform are then quantised. This leads to visually significant coefficients being retained and insignificant coefficients being discarded, further improving video compression. The trade-off, however, is that once coefficients are discarded, the reconstruction of the image to the original

is impossible but if done right, a visually similar image can be reconstructed (Richardson, 2002).

Once the energy compaction stage is complete, the symbol compaction stage is next. This stage is also referred to as entropy coding. The entropy encoder is responsible for representing each quantised coefficient with as short a binary codeword as possible. Methods which are used for this module include Huffman encoding and Context-adaptive binary arithmetic coding (CABAC) (Marpe et al., 2003). Both of these take advantage of statistical redundancies in the bitstream to achieve further compression. This is a very high-level view of a video codec, and there are many underlying sub-modules within modern video codecs which are detailed in a later chapter.

1.2 Improving Video Codecs

In the last five years, there has emerged significant work in using deep learning for video codecs (Chen et al., 2021; Lu et al., 2020). Examples include replacing components of HEVC and VVC with deep learning tools (Liu et al., 2019) which set the framework for which the Deep Learning Video Codec (DLVC). The DLVC has a CNN-based in-loop filter and CNN-based block adaptive resolution coding (Liu et al., 2020). Similarly, the MOVI-codec was designed to replace any motion estimation or motion compensation modules within a codec with a deep network (Chen et al., 2021). The major drawback of these methods is the high computational load required and that commonly used hardware for video streaming (i.e. mobile phones) is only very recently capable of supporting deep networks within video compression codecs (Hou, 2021). Because of this, we will not focus on the innovations of deep learning based video codecs, but on how to improve well-established video codecs.

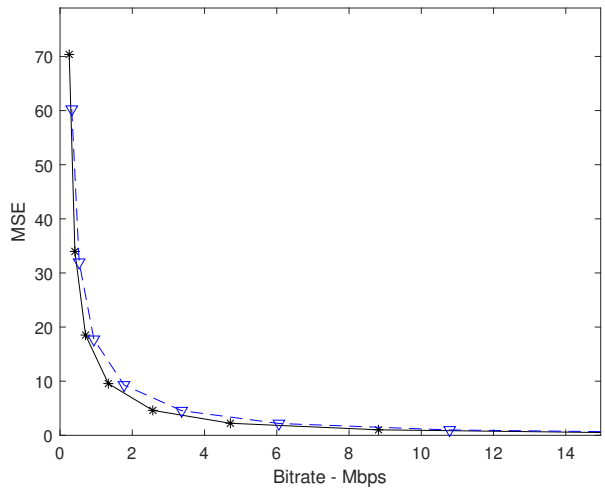

Fig. 1.2 *The performance of two systems are being compared through their Rate-Distortion (RD) Curves. The RD curve (black) closer to the bottom left of the graph corresponds to the better-performing system. That indicates low distortion at a lower bitrate than the blue curve.*

1.2.1 Measuring Performance

It is crucial to quantify the improvements made to video encoders. The quality of a video can be measured subjectively using tools such as the double stimulus continuous quality scale (DSCQS) (BT, 2002) or objectively using metrics such as Peak Signal to Noise Ratio (PSNR), Structural Similarity Image Metric (SSIM) (Wang et al., 2004) and Visual Multimodal Assessment Fusion (VMAF) (Blog, 2016).

We can visually represent the effectiveness of video compression by using a Rate-Distortion (RD) Curve. In practice, it is common for a Rate-Quality (RQ) curve to be used and referred to as an RD Curve. For example, we see in Satti et al. (2019); Wu et al. (2021); Zhang and Bull (2019) and others, the use of quality metrics such as PSNR, SSIM and VMAF on the y-axis of their presented graphs labelled as RD Curves. There is a link between quality and distortion as the most common quality metric used (PSNR) is a function of the distortion metric Mean Squared Error (MSE). We can see in Figure 1.2 that there is a relationship between distortion and bitrate.

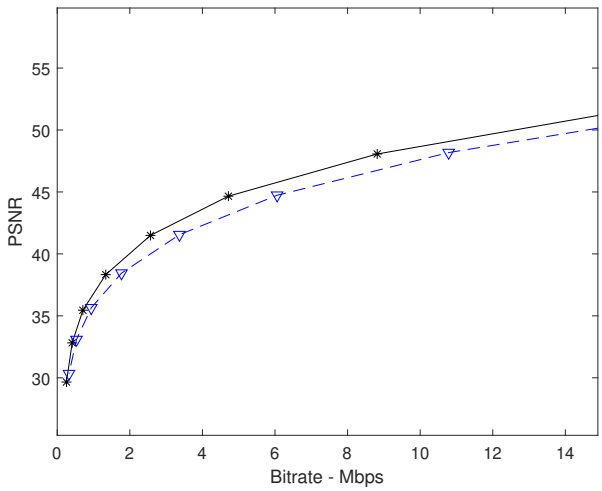

Fig. 1.3 *The performance of two systems is being compared using the quality metric PSNR. The RD curve (black) closer to the top left of the graph corresponds to the better-performing system. That indicates high quality at a lower bitrate than the blue curve. Comparisons are typically made using RQ-curves in the field of video compression instead of using distortion on the y-axis*

On this plot, bitrate is shown on the horizontal axis while the vertical axis shows the distortion metric. Each curve point is generated by invoking the encoder with a particular parameter setting. As that parameter is changed, the measured R/D combination of the output maps out a curve.

However, quality metrics such as PSNR are more commonly used than distortion metrics such as MSE and there is a community knowledge of what PSNR values are expected to represent a good picture or video, and what PSNR values are expected to represent a bad picture or video. Because of this knowledge, it is more common to see Rate Quality curves. The remainder of this thesis will be using the convention of using a quality metric on the y-axis of an RD-Curve.

In the example, in Figure 1.3 PSNR is used as the quality metric. As a higher quality video, at a lower bitrate is preferred, the ideal case is a curve as close to the top left corner as possible. We can also use RD-Curves to quickly compare compression schemes. Figure

1.3 which shows two RD-Curves, it can be seen that the black solid curve has a better performance than the blue dashed curve.

In order to quantify the difference between two codecs, or the same codec under different parameter settings, Bjøntegaard Delta Rate (BD-Rate) (Bjøntegaard, 2001) is used. The BD-Rate, B_r, is defined as the area between these curves. This is the typical metric used to measure compression impact and is expressed as follows.

$$B_r = \frac{1}{D_2 - D_1} \int_{D_1}^{D_2} (R_A(D) - R_B(D)) dD$$

Where the integral is evaluated over the quality range $[D_1, D_2]$ for the two curves R_A and R_B. A negative BD-Rate indicates that we have achieved the same quality at a lower bitrate i.e. we are closer to the top left of the graph. It is possible to use either BD-Rate or BD-Distortion to compare performance. In this thesis, we will focus on delivering equal quality at a lower bitrate, and will be primarily using BD-Rate.

1.2.2 Re-engineering Codec Tools

While there is work on deep learning based video codecs, the well-established codecs (MPEG and AOM codecs) are here to stay (Laude et al., 2019). In popular video on-demand services, older established video codecs are still deployed. In 2016, over 70% of uploaded videos to YouTube were H.264 (Kokaram et al., 2016) and in 2020, Netflix reported that H.264 devices are a "substantial portion of the members viewing hours and a large portion of the traffic" (Afonso et al., 2020) as there are still user devices in use which are not capable of decoding the bitstreams compliant with the latest compression standards. Since these standards define the decoder and the syntax of bitstream and not the encoder, it is still possible to improve the performance of the encoding process long after the standardisation effort is complete. Because of the additional pressures on internet distribution and globally

available bandwidth, there has always been an effort to re-engineer existing codecs to hit better rate/quality tradeoffs.

For example, the first version of H.264 was standardized in 2003 (Wiegand et al., 2003) and since then thousands of citations to publications improving the usefulness of the tools have emerged. In 2007 the work by Merritt and Vanam (2007) improved the rate control algorithm beyond the reference model and that enabled H.264 to be used for internet distribution by YouTube, Vimeo and others. This development continues today, almost 20 years later (Bernatin et al., 2021). This applies not only to H.264, but H.265 and VP9 as well.

There are a host of tools implied in video encoding/decoding workflows which have also benefited from re-engineering over the years. Skupin et al. (2021) improved the selection of the Group of Picture block. The decoder inside the codec is an important candidate for re-engineering and Tu et al. (2021) built a better quality predictor in that module performing 20× faster than their baseline. The in-loop filter has also been addressed e.g. in HEVC (Kuanar et al., 2018). Motion estimation is a very long standing research topic overlapping with Optical Flow research in the computer vision community and it is at the heart of the frame prediction process inside the encoder. A limited number of research groups have evaluated the use of more modern motion estimation tools in the motion compensation blocks (Chi et al., 2007; Kameda et al., 2016b; Young and Taubman, 2015b).

Re-engineering the content delivery system has also resulted in great impact well after standardisation. Videos are rarely encoded end to end in content delivery systems, but in short chunks in order to provide users with a seamless viewing experience regardless of their device's network conditions. This concept is called Adaptive Bitrate Streaming (ABR) and is a primary system component in streaming services today (Bentaleb et al., 2018; Katsenou et al., 2019). There are two competing standards in this area, DASH (Dynamic Adaptive Streaming over HTTP) (Sodagar, 2011) and HLS (HTTP Live Streaming) (Pantos and May, 2017). In 2015 Netflix (Aaron et al., 2015) disrupted ABR technology by proposing that each of the chunks used in HLS or DASH could be optimised on a per-chunk basis rather

than using the same parameter set for encoding across the whole clip. This led to significant gains in both bitrate and quality and set a benchmark for performance ever since.

With the optimization effort over the past decades, now it is becoming very challenging to be able to achieve even a small coding gain over H.265/HEVC. For example, in some recent publications in the IEEE Journals and Conferences in the field of Signal Processing, 1% or 2% of average improvement over H.265/HEVC are considered significant (Baker et al., 2011; Kuanar et al., 2018; Tu et al., 2021; Young and Taubman, 2015b; Zhang and Bull, 2019). Nonetheless, due to the scale of video streaming and the continuing pressure on computation and bandwidth, many research organizations are still actively working on improving the coding performance. The historical expectation is that accumulation of small improvements will eventually result in significant system improvements overall especially as the scale of deployment is consistently growing.

These are just a few examples within a very broad field of video compression. Given the importance of motion in the hybrid encoding model and the relatively few prior efforts addressing this tool, in the next section we present a case study in re-engineering the motion estimator. Without affecting the bitstream syntax we replace the classical block matcher with modern optical flow techniques. This work was published in **"Using modern motion estimation algorithms in existing video codecs."** by Daniel J. Ringis, Davinder Singh, François Pitié and Anil Kokaram, *in Applications of Digital Image Processing XLI* (Vol. 10752, pp. 288-295). SPIE. September, 2018.

1.2.3 Re-engineering Codec Tools: Case Study Using Motion Vectors

The inter-frame prediction module in a hybrid encoder is key to compression of video sequences. A hybrid encoder is a video encoder which has decoding elements within it. The motion compensation module typically uses a block matching motion estimation algorithm to associate blocks across frames. It is well known that block matching suffers from some serious limitations with respect to the estimation of the true underlying motion in a video

sequence. The development of new motion estimation techniques is a huge area of research both in computer vision and signal processing. It makes sense then to test the notion of replacing the block matcher (first proposed in the 1970's (Candy et al., 1971)) with these more modern techniques developed more than 40 years later.

We tested DisFast (Kroeger et al., 2016) and DeepFlow (Weinzaepfel et al., 2013). DisFast used a Dense Inverse Search to quickly find corresponding patches. DeepFlow was one of the first Deep Learning based optic flow estimators. The work was initially performed on two video codecs, MPEG4 and H.264, which were some of the most widespread at the time of experiments. The engineering effort involved in these tests are not trivial. The key complication in replacing the motion estimator is that in a hybrid encoder, motion vectors are generated from a simulated decoding process within the encoder. In detail therefore the systems for motion estimation have to be integrated into the encoder software model which itself is quite complex. An in depth description of this work can be found in Appendix 1.

There were a number of different algorithms explored for combining better motion estimators with existing block matching. However the core experiment was to test the use of ground truth motion and evaluate the impact, then test the impact when that was not available. Using the SINTEL dataset (Butler et al., 2012) and associated ground truth motion we observed 1.5% and 1.6% average improvement for MPEG4 and H.264 codecs respectively. When using estimated motion with DisFast, we observed an average BD-Rate improvement 1.53% and 1.63% improvement with MPEG4 and H.264 respectively. The best BD-Rate improvement observed across the dataset used was 4.4% in MPEG4 and 3.4% in H.264 using DisFast.

These results showed first that the potential improvement is not insignificant, which is encouraging. However there was little difference in impact between ground truth and apparent motion. That implied that the incorporation of the rate/distortion error criterion for selecting motion was playing a large role in the selection of appropriate prediction.

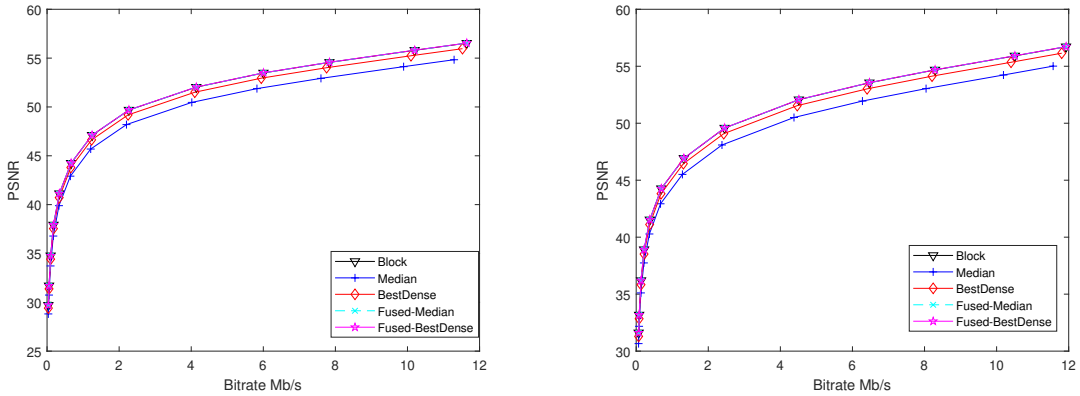

Fig. 1.4 *Aggregated RD-Curves for the live action sequences (720p) encoded using different motion estimators for VP9 (left) and H.265 (right). If the insertion of the optical flow vectors led to improvements over Block Matching, the RD-Curves of* `Median`*,* `Fused-Median`*,* `BestDense` *and* `Fused-BestDense` *systems would be closer to the top left of the graph compared to* `Block`*. However we can see that the block matcher performed as good as the optical flow vectors with very little difference between the RD-Curves for both codecs. On average we measured a 0.3% improvement. Please see Appendix 1 for more detailed explanation.*

Also interesting was the fact that with the more modern codecs VP9 and H.265 we measured just 0.3% average improvement when using DisFast as compared to using Block Matching for both codecs, with 0.77% (H265) and 0.89% (VP9) best improvement. It seemed therefore that other tools, and indeed the RD optimisation tools in those codecs were having a larger impact than the motion estimator. Figure 1.4 visualises the RD curves for various different algorithms that combined block matching with DisFast. Clearly the performance of just the default block matcher, as an average over the whole dataset was still good compared with the more modern motion estimators. This observation led us to focus instead on the RD modules to possibly extract more significant gains. Please refer to Appendix 1 for details of experiments, datasets and a more complete analysis of algorithms.

1.3 Optimising the RD Trade-Off

Ideally, a compressed video would have zero distortion at as low a bitrate as possible, but we know that is not always practical. So decisions to achieve the optimal trade-off between bitrate and distortion need to be made.

One such method of determining the ideal balance between bitrate and distortion is the Lagrange multiplier approach which was advocated by Sullivan and Wiegand (1998). This continues to be the adopted approach in video codecs since 1998 (Zhang and Bull, 2019). In this approach the codec makes a number of decisions in order to minimise a cost J as follows,

$$J = D + \lambda R \tag{1.1}$$

where D is the Distortion, R is the Rate and λ is the Lagrangian multiplier. λ is selected to allow for different operating points in the rate-distortion curve. Other options are selected to minimize J for a given lambda (Sullivan and Wiegand, 1998). Adjusting λ in this equation can lead to a significant change in video encoder performance. Estimation methods for the Lagrangian Multipler, λ, deployed in H.264/AVC and HEVC reference encoders used empirically derived values (Sullivan and Wiegand, 1998). By testing a range of λ over a small cohort of five(5) sequences an optimal value was estimated.

Clearly this is a limited dataset and the content in 1998 does not reflect the video content available twenty five years later e.g. resolution and bit depth have changed. In addition, optimisation over a corpus suppresses gains for individual clips. Optimisation of the Lagrangian multiplier, λ, is therefore an important part of the video encoding process and hence in Chapter 2 we detail its history and development.

1.3.1 Per Clip Optimisation

Streaming services, such as Netflix, have relatively small libraries of videos which they wish to distribute to users at high quality and low bitrate (Aaron et al., 2015; Katsavounidis and

Guo, 2018) recognised that by choosing codec parameters *per clip*, significant gains could be achieved. This is because the statistics of video clips vary greatly over any corpus. Hence tuning for average improvement does not achieve the best gain possible for each clip. While that work considered only the design of a bitrate ladder for DASH streaming, it provided an impetus for *per clip* optimisation throughout the codec system. Netflix refered to this as per title encoding approach (Aaron et al., 2015). Covell et al. (2016) also built on that idea by optimising other codec parameters (CRF) in that case for *per clip* transcoding.

In addition to the setting of external parameters when encoding video, it is possible that re-engineering parameters or tools which are within the codec structure can lead to improvements in compression. These parameters are often overlooked. Therefore we can adopt a similar approach to a video, not just in the bitrate ladder, but in terms of choosing the rate distortion multiplier, λ, that adapts to each clip individually.

1.3.2 Re-Engineering the Lagrangian Multiplier in Video Codecs

As discussed previously, the Lagrangian mulitplier (λ) as used in existing codecs is likely to be sub-optimal as it has not been fully validated using typical video content (Zhang and Bull, 2019). It is therefore important to explore the potential improvements which can be found by re-engineering the Lagrangian multiplier appropriately for modern video content. In this section, we will briefly highlight some different implementations of re-engineering λ within H.264, H.265 and VP9. A more thorough review of the works mentioned here will follow in Chapter 2.

Zhang and Bull (2019) determined that λ should be calculated and adjusted based on the distortion ratio of B Frames and P Frames. This work determined that for B-frames, the default value for λ (denoted as λ_o) calculated in the codec is not optimal for every type of video. Instead they modify the λ used away from λ_o using a simple constant of proportionality $\lambda = k\lambda_o$. While not explicitly stated in the reviewed literature, this method of adjusting the current implementation of λ appears in other works. Papadopoulos et al. (2016)

had a very similar approach but applied an offset to Q which is directly linked to λ in codec implementations.

Another interesting approach to adjusting λ based on the video content was to use video features to determine a multiplier for λ (Ma et al., 2016). The video features used were Spatial Information and Temporal Information. Using these metrics, a Support Vector Machine (SVM) was used in deciding the value of k. Hamza et al. (2019) also take a classification approach but using gross scene classification into indoor/outdoor/urban/non-urban classes. They then used the same k for each class. Yang et al. (2017) used a combination of features instead of just the Mean Squared Error (MSE) ratio defined in (Zhang and Bull, 2019). In their work, they used a perceptual content measurement to model k. Again, each of these works have a similar core approach, that is to re-engineer the Lagrangian multiplier with some form of offset or proportionality constant.

The equation $\lambda = k\lambda_o$ is the center of this work. We determine k for implementation within existing Lagrangian multiplier schemes within the HEVC and VP9 codecs. This concept is expanded in Chapter 3.

1.3.3 Key Observations and Motivation

The existing work has overlooked the possibility that a direct search using BD-Rate as the objective function, could lead to even more improvements. In some way all these previous works have reported data showing k versus $BD-Rate$, but they use various summary models to simplify this relationship across a corpus. By using instead Direct optimisation, we are able to explore just how much more BD-Rate gains there are per-clip.

The size of the corpus used in previous work was quite small (e.g. 37 clips in Ma et al. (2016) and 36 in Zhang and Bull (2019) up to 300 frames per clip) and the types of content used in previous corpora are not necessarily a good representation of modern material. Exploring an improved rate distortion performance for modern user generated content is

important at a large scale to explore the potential impact in a real world application of this work. These observations lead us to our research questions as follows.

Research Question 1: What are the potential gains achievable by per clip optimisation of the Lagrangian multiplier compared to the current implementation in common video codecs?

Research Question 2: Can we achieve the potential gains at a reduced computational load through the use of proxy systems or prediction ?

1.4 Contributions

This work investigates the impact of re-engineering a single internal parameter, the top level Lagrangian multiplier, on a per clip basis. By re-engineering on a per clip basis we hope to find the maximum improvement a clip can have when the Lagrangian multiplier is tailored to that specific video, and subsequently determine if there are any models which we can use to replicate that improvement as efficiently as possible. This leads us to the following contributions:

Contribution 1: Implementation across a large corpus. Prior work was implemented on standard datasets or corpora of up to 100 video clips. Our work used approximately 10,000 video clips from standard video datasets for video compression.

Contribution 2: BD-Rate improvement of up to 25% in a single clip using Direct Optimisation. The average BD-Rate improvement was approximately 2% across the 10k clip corpus.

Contribution 3: Approximately 60% of the possible BD-Rate improvements were achieved by direct optimisation using proxies or prediction. This effectively reduced the computational complexity ten fold.

In this thesis, we will focus on High Efficiency Video Coding (HEVC/H.265) and VP9 codecs. At the time this work begun, these were the newest widely adopted codecs (Timmerer

et al., 2021). There are newer established codecs, such as AV1 or VVC but these will not be considered.

1.4.1 Publications

In support of this thesis the following conference papers and presentations were submitted, accepted and published.

- **"Using modern motion estimation algorithms in existing video codecs."** by Daniel J. Ringis, Davinder Singh, François Pitié and Anil Kokaram, *in Applications of Digital Image Processing XLI* (Vol. 10752, pp. 288-295). SPIE. September, 2018.

  This paper was our first investigation into re-engineering the tools in video codecs. We initially focused on the motion estimation tools within the codecs of MPEG-4 and H.264. Upon further investigation of the viability in H265 and VP9, there were negligible improvements which led to experiments on other codec sub-modules such as the Rate- Distortion Optimisation module.

- **"Per Clip Lagrangian Multiplier Optimisation for HEVC."** by Daniel J. Ringis, François Pitié, and Anil Kokaram, in *Electronic Imaging 2020*. Electronic Imaging. January, 2020.

  This paper investigates the impact of re-engineering the rate distortion equation on compression performance. A constant of proportionality, k, is used to modify the Lagrange multiplier used in H.265 (HEVC). Direct optimisation methods are deployed to maximise BD-Rate improvement for a particular clip.

- **"Per-clip and per-bitrate adaptation of the Lagrangian multiplier in video coding."** by Daniel J. Ringis, François Pitié, and Anil Kokaram, in A*pplications of Digital Image Processing XLIV,* vol. 11842, pp. 185-194. SPIE, 2021.

  Here we present a framework for choosing the best Lagrangian multiplier on a per-operating point basis across a range of bitrates. In effect, we are trying to find the

para-optimal gain across bitrate and distortion for a single clip. We optimize both for bitrate savings as well as distortion metrics (PSNR, SSIM).

- **"Per-clip adaptive Lagrangian multiplier optimisation with low-resolution proxies."** by Daniel J. Ringis, François Pitié, and Anil Kokaram, in *Applications of Digital Image Processing XLIII*, vol. 11510, pp. 40-51. SPIE, 2020.

  This work focuses on reducing the computational cost of repeated video encodes by using a lower resolution clip as a proxy. Features extracted from the low resolution clip are used to learn the mapping to an optimal Lagrange Multiplier for the original resolution clip. In addition to reducing the computational cost and encode time by using lower resolution clips, we also investigate the use of older, but faster codecs such as H.264 to create proxies.

- **"Near optimal per-clip Lagrangian multiplier prediction in HEVC."** by Daniel J. Ringis, François Pitié, and Anil Kokaram, in *2021 Picture Coding Symposium (PCS)*, pp. 1-5. IEEE, 2021.

  A key component of these algorithms is modeling the R-D characteristic across the appropriate bitrate range. This is computationally heavy as it usually involves repeated video encodes of the high resolution material at different parameter settings. This work focuses on reducing this computational load by deploying a Neural Network (NN) operating on lower bandwidth features. This work received a **Top 10 Paper** award.

1.5 Thesis Outline

The rest of this thesis is structured as follows.

Chapter 2: **Literature on Rate Distortion Optimisation in Video Codecs** of comparable work in the field of per clip video encoding and Lagrangian Multiplier Adjustments. In particular, a detailed look at how the Lagrangian Multiplier is derived and implemented in

video codecs and a review of other work which applies per clip adjustments of the Lagrangian Multiplier through either classification, regression or quantiser manipulation.

Chapter 3: **Re-engineering Rate Distortion Optimisation through Direct Optimisation** In this chapter we investigate the impact of re-engineering the rate distortion equation on compression performance. A constant of proportionality, k, is used to modify the Lagrange multiplier used in H.265 (HEVC) and VP9. Direct optimisation methods are deployed to maximise BD-Rate improvement for a particular clip.

Chapter 4: **Per Operating Point Rate Distortion Optimisation:** In this chapter, we present a framework for choosing the best Lagrangian multiplier on a per-operating point basis across a range of bitrates. In effect, we are trying to find the pareto-optimal gain across bitrate and distortion for a single clip. We employ direct search techniques to address this optimisation problem.

Chapter 5: **Reducing Computational Cost with Proxies:** This chapter focuses on reducing the computational cost of repeated video encodes by using a lower resolution clip as a proxy. Features extracted from the low resolution clip are then used to learn the mapping to an optimal Lagrange Multiplier for the original resolution clip. In addition to reducing the computational cost and encode time by using lower resolution clips, we also investigate the use of older, but faster codecs such as H.264 to create proxies.

Chapter 6: **Reducing Computational Cost through Prediction:** This chapter looks at modeling the R-D characteristic across the appropriate bitrate range. This is computationally heavy as it usually involves repeated video encodes of the high resolution material at different parameter settings. This work focuses on reducing this computational load by deploying a NN operating on lower bandwidth features.

Chapter 7: **Conclusions:** The final Chapter assesses the contributions of this thesis, considers the Computational Infrastructure and Environmental Impact of this work and the potential directions of future work.

Chapter 2

Literature on Rate Distortion Optimisation in Video Codecs

Within a video codec there are a number of processes and modules which have the same common goal, "minimise bitrate and maximise quality". The video coding system needs to make a decision for many parameters for each frame and each macroblock. In HEVC for example, these include Intra/Inter prediction modes, CTU/MB segmentation, and quantization step sizes. All of these decisions contribute to the shared goal of minimising distortion (i.e. maximising quality) while also keeping the bitrate low.

The task of rate distortion optimisation is to choose for each image region, the parameter set which leads to the most efficient coded representation. This includes making the appropriate decisions that affect segmentation, prediction modes, motion vectors and quantization levels for example (Sullivan and Wiegand, 1998).

2.1 Rate Distortion Optimisation in Video Compression

Rate distortion optimisation can either represent the minimisation of the bitrate, R, while keeping the distortion, D, below a set target, $D_{\max}$, or, alternatively, the minimisation of the distortion, D while keeping the rate below a target bitrate, $R_{\max}$. The optimisation is based

on tuning factors including quantisation step-size, QP and macroblock type selection, M. In the following, we will consider the latter optimisation problem, that is:

$$\begin{aligned} &\underset{QP,M,\ldots}{\text{minimize}} \quad D(QP,M,\ldots) \\ &\text{subject to} \quad R(QP,M,\ldots) < \mathrm{R}_{\max}. \end{aligned} \tag{2.1}$$

This problem can be solved using Lagrangian optimization (Everett III, 1963; Ortega and Ramchandran, 1998; Sullivan and Wiegand, 1998). In this approach the codec transforms the constrained optimisation problem into an unconstrained problem by introducing the cost function J from Equation 1.1, repeated below.

$$J = D + \lambda R$$

J combines both the distortion D (for a frame or macroblock) and the rate R (the number of coded bits for that unit) through the action of the Lagrangian multiplier λ.

This technique was adopted as it is effective and conceptually simple. For each value of λ, minimising J w.r.t. Quantiser Parameter, QP yields an optimal solution to Eq. (2.1) for a certain $R_{\max}$. Conversely for any $R_{\max}$, there exists an optimal value of λ.

However, in practice, there are a number of interactions between coding decisions which make this problem less straightforward (Ortega and Ramchandran, 1998).

2.1.1 Linking the Lagrangian Multiplier and Quantiser Parameter

The first issue is how to find the optimal value of λ for a given targeted rate. The seminal work of Sullivan and Wiegand (1998) laid the foundation for an experimental approach to choosing λ.

There is a typical high rate approximation in signal compression (Jayant and Noll, 1984) as follows:

$$R(D) = a \ln\left(\frac{b}{D}\right) \tag{2.2}$$

where a and b are two parameters that vary depending on the video content. Substituting this into Eq. (1.1) yields the following equation:

$$J = D + \lambda a \ln\left(\frac{b}{D}\right)$$

Taking J as a function of D, we have at the minimum

$$\frac{dJ}{dD} = 1 - \lambda a \frac{1}{D} = 0,$$

thus

$$\lambda = \frac{D}{a}. \tag{2.3}$$

From 2.2 and 2.3 we also have:

$$\frac{dR}{dD} = -\frac{a}{D} = -\frac{1}{\lambda},$$

Here we note that

$$\lambda = -\frac{dD}{dR} \tag{2.4}$$

which means the optimal Lagrangian multiplier is the negative slope of the curve R vs D. This is illustrated in Figure 2.1.

Assuming the quantiser noise is uniformly distributed, the relationship between distortion power and quantiser step size is as follows (Proakis, 2000).

$$D(Q) = \frac{QP^2}{12} \tag{2.5}$$

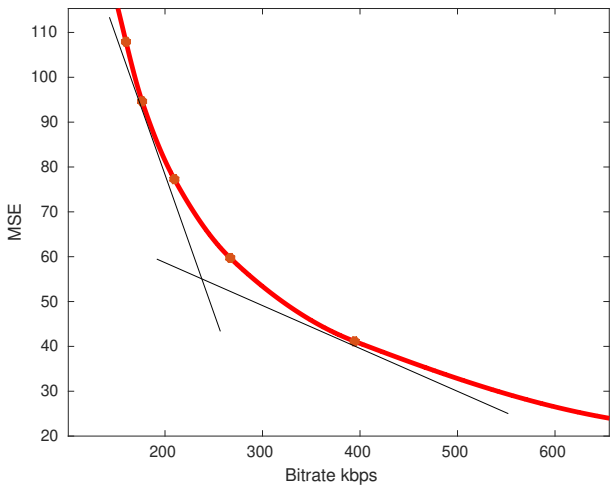

Fig. 2.1 *RD Curve (red) for a video sequence (foreman) encoded using x263. This figure is a zoomed view of the curve determined by using the operating points QP = 7 : 5 : 32. If all the assumptions that led to equation 2.4 were true, then the gradient of the line tangent to the curve at a particular RD-point (black lines) is the value of λ, which minimises J for that operating point.*

Substituting for D from this equation into Equation 2.2 yields a relationship between R and QP as follows

$$R(QP) = a\ln\left(\frac{12b}{QP^2}\right) \tag{2.6}$$

Taking derivatives from Equation 2.5 and 2.6 yields a relationship between λ and QP as below:

$$\lambda = -\frac{dD}{dR} = -\frac{dD/dQ}{dR/dQP} = -\frac{QP/6}{-2a/QP}$$

which simplifies to

$$\lambda = \frac{1}{12a} \times QP^2. \tag{2.7}$$

In 1998 a was determined for H.263 empirically based on experiments conducted with a sample video set (Sullivan and Wiegand, 1998). The section 2.1.3 details this procedure.

2.1.2 Lagrangian Multiplier in Early Video Codecs

There are many coding decisions in the encoder which are dependent on Rate and Distortion. Hence, these coding decisions can also be made with the help of Lagrange Optimisation. For example, the bitrate of a video compressed by a codec is directly impacted by the prediction mode decisions, motion vector (MV) choices and the coding fidelity of the direct frame difference (DFD). The coding fidelity is controlled by choosing a step size to be used for quantization and reconstruction of the transformed and compressed signal. A large quantizer step size leads to a lower bitrate and a large amount of distortion.

In addition to the bitrate control, motion estimation and in particular the associated motion vector decisions are vital in the Rate Distortion Optimisation process. In video codecs, motion estimation is performed by searching for the motion vector which minimises the prediction error for a pair of frames, with some preference for motion vectors which require the fewest bits to encode (Sullivan and Wiegand, 1998). As with the overall bit rate control, motion vector searches and coding are linked to the trade-off between rate and distortion. Within a codec, the error criterion used with motion estimation can be viewed as a Lagrangian cost function:

$$J_{\text{MOTION}} = D_{\text{DFD}} + \lambda_{\text{MOTION}} R_{\text{MOTION}} \tag{2.8}$$

where the distortion D_{DFD} represents the prediction error of the direct frame difference and R_{MOTION} is the number of bits required to represent a given motion vector. The Lagrange Multiplier λ_{MOTION}, imposes the weighted decision between Rate and Distortion in Motion Vector decisions (Girod, 1994).

With motion vectors having an impact on rate distortion optimisation, it is natural that the macroblock (MB) or coding tree unit (CTU) sizes are affected. In addition to the size of the macroblocks, the mode of macroblock, whether it is coded using SKIP, INTRA or

Fig. 2.2 *Frames from the four sequences used to establish a link between λ_{MODE}, λ_{MOTION} and QP.*

INTER modes also impacts the rate distortion optimisation of a video codec. These can be represented using another Lagrangian equation as follows.

$$J(M,Q) = D_{\text{REC}}(M,QP) + \lambda_{\text{MODE}}\, R_{\text{REC}}(M,QP) \tag{2.9}$$

In this equation, M is the type of coding mode, QP is the Quantizer step size, $D_{\text{REC}}(M,QP)$ is the error between the original uncompressed macroblock and the reconstruction and $R_{\text{REC}}(M,QP)$ is the number of bits associated with the given macroblock (Schuster and Katsaggelos, 1996).

2.1.3 Determination of the Lagrangian Multiplier in Video Codecs

For the H.263 Video Compression Standard, Sullivan and Wiegand (1998) and Ortega and Ramchandran (1998) experimentally determined that the implementation of Equation 2.7 should be $\lambda = \lambda_{\text{MODE}} = 0.85 \times QP^2$.

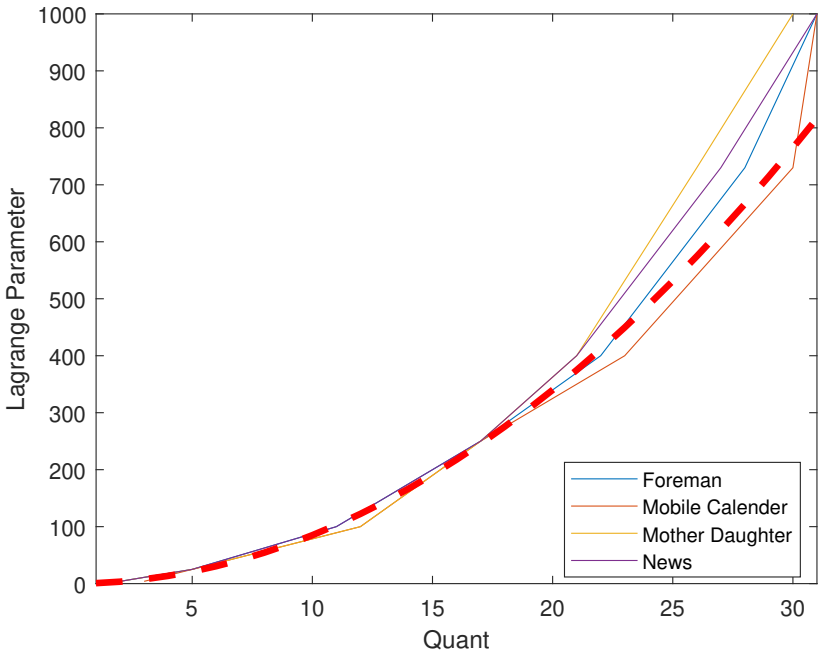

Fig. 2.3 *Quantiser step size vs Lagrangian Parameter for four clips and the approximation* $\lambda = 0.85 \times QP^2$ *shown in red, recreated from Sullivan and Wiegand (1998). One of the main points of this dissertation is that this fit is calculated using only four (4) videos and there is still a fair amount of deviation in these examples, particularly in the higher QP range.*

In order to determine this relationship, they set the Lagrange parameter λ_{MODE} to seven different values (4, 25, 100, 250, 400, 730 and 1000) and encoded 100 frames of four sequences. These sequences were *Mother and Child, Foreman, Mobile and Calender and News* (available on Xiph.org (2018)). Frames from these sequences can be seen in Figure 2.2. For each sequence and each λ combination they estimated the optimal QP which minimised J over each macroblock. The curve of λ vs QP_{opt} is approximated by the function $\lambda = 0.85 \times QP^2$ as shown in Figure 2.3 (Sullivan and Wiegand, 1998). Similarly, using these same four video clips, it was determined that $\lambda = \lambda_{\text{MODE}} = \sqrt{\lambda_{\text{MOTION}}}$. Given QP, their relationship allowed these parameters to be implemented in H.263.

In future video codecs, H.264 and H.265, bi-directional frames (B-Frames) were introduced and the experiments to establish links between λ_{MODE}, λ_{MOTION} and QP were repeated. Three different relationships were established for each of the Intra (I), Predicted

(P) and B frames as follows (Sullivan et al., 2012).

$$\lambda_I = 0.57 \times 2^{(QP-12)/3} \tag{2.10}$$

$$\lambda_P = 0.85 \times 2^{(QP-12)/3} \tag{2.11}$$

$$\lambda_B = 0.68 \times \max(2, \min(4, (QP-12)/6)) \times 2^{(QP-12)/3} \tag{2.12}$$

It is unclear what dataset was used to establish these relationships as no public validation was available (Zhang and Bull, 2019) but the methodology was the same as deployed for H.263 in 1998. Nevertheless these updated relationships were recommended for use with H.265.

In VP9, λ was estimated by optimisation over a different/modern test video clip set (Mukherjee et al., 2013). Hence λ is estimated from q_i (the quantiser index in VP9) as follows:

$$\lambda = q_{dc}^2 \times (A + 0.0035 \times q_i), \tag{2.13}$$

where A is a constant depending on frame type ($3.2 \leq A \leq 3.3$) and $q_{dc} = f(q_i, A)$ is another defined mapping implemented with a discrete valued LUT.

Each of these implementations perform reasonably well of course, however there is potential for improvement. Firstly, these empirical relationships perform well *on average* but for any *particular clip*, the empirical estimate is unlikely to be optimal. Secondly, the size of the corpus used for these experiments was quite small, so there is room for exploration with a larger corpus.

In the next section, we look at the how the tailoring of coding parameters based on the video content has been beneficial to compression performance. In particular, we will be looking at work which focuses on adjusting the Lagrangian multiplier directly, or the quantiser parameter to indirectly adjust λ.

2.2 Per Clip Codec Parameters Optimisation

We recognise that the Lagrangian multiplier is not optimal across all videos. For other parameters in video encoding, the idea of adjusting based on the video content was established by Aaron et al. (2015) and generalised by Katsavounidis and Guo (2018). They referred to it as "Per Clip Encoding".

In their case, they considered the bitrate ladder, and recognised that the one size fits all approach was sub-optimal (Aaron et al., 2015; Katsavounidis and Guo, 2018). We repeated the experiment reported from their original paper (Aaron et al., 2015) with HEVC and results are shown in Figure 2.4. We clearly see the high diversity in output RD-performance across six titles when encoding with the same set of parameters. The point is that for some clips with complex content the output quality remains low, even at high bitrates (e.g. rushfield), whereas for some simple content (e.g. sunflower) the quality peak is achieved at a very low bitrate. This led to the idea that tailoring the bitrate ladder to a specific video clip would lead to significant improvements.

Based on this, a per-title approach was established. Aaron et al. (2015) searched for the optimal bitrate ladder for each of the video clips being encoded. Using this approach, Katsavounidis and Guo (2018) was able to achieve 20% bitrate savings at the same quality as the previous approach. As this was intended for a relatively small video corpus, they were able to scale their optimization to all the titles in Netflix's library. This is suitable for a relatively small corpus of professional videos, but may not be fitting for larger video corpus.

For a large corpus of user generated video content, YouTube established a deep learning based system which used features of the video to determine the recommended encoding parameters to achieve a target bitrate for a given video. In that work they considered estimating Constant Rate Factor (CRF) optimally (Covell et al., 2016). Almost 10,000 five second video clips were used, which is a significantly larger dataset than previously employed in these sorts of experiments. Features from these clips were used as the input vector for a Neural Network. The features were, average number of bits used for the motion

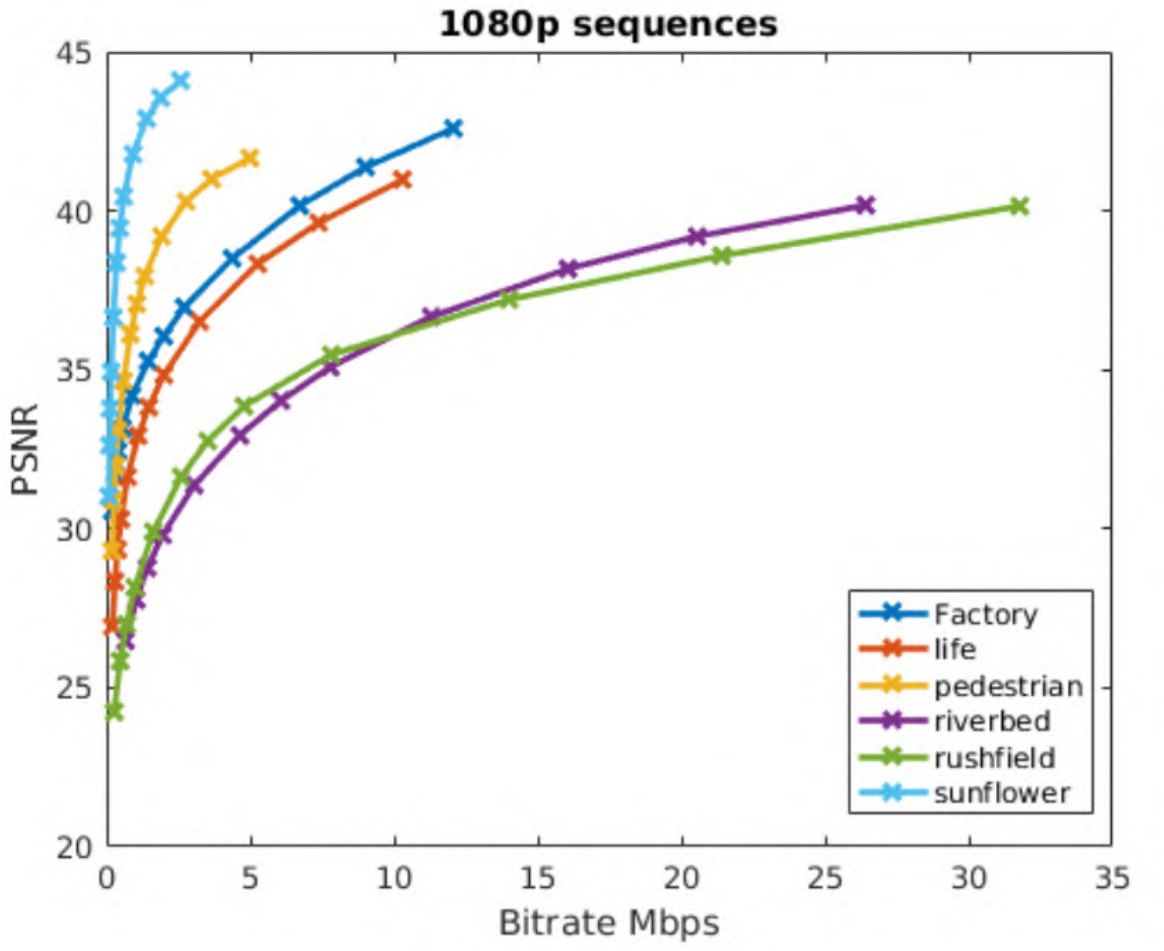

Fig. 2.4 *RD-Curves for different 1080P sequences from Xiph.org (2018), encoded using identical parameters. The plots show that some titles reach very high PSNR (45 dB or more) at bitrates of 2500 kbps or less. On the other extreme, some titles require bitrates of 30 Mbps or more to achieve an acceptable PSNR of 38 dB. Based on the idea put forward by Aaron et al. (2015)*

vector (MV) per predicted (P) macroblock (MB), average number of bits used for the texture (i.e., pixel values or prediction error) per MB, average number of bits used for the texture per intraframe (I) MB, average number of bits used for the texture per P MB, percentage of I MBs, percentage of skipped MBs, a score on the complexity of the video encoding for this segment, average quantization-parameter setting used in the transcode. They were able to predict the CRF accurately enough to achieve the target bitrate within 20% error for 80% of their test set.

As an extension of the work done by Katsavounidis and Guo (2018), Amirpour et al. (2021) improved the bitrate ladder on a per title basis. The optimisation parameter were frame-rate and resolution. Hence, their bitrate ladder allowed for variable frame-rate and non-canonical resolutions. They were able to achieve a 13% BD-Rate improvement over the HEVC Reference Codec parameters on the UltraVideo dataset (Mercat et al., 2020).

Similarly, Satti et al. (2019) proposed a per clip approach which was based on a model linking the bitrate, resolution and quality of a video. Using this model allowed for a bitrate ladder to be generated for a given clip at a lower computational cost than Katsavounidis and Guo (2018).

Wu et al. (2021) built on the framework of the Dynamic Optimiser (Katsavounidis and Guo, 2018) by deploying a cross-codec approach which was capable of matching the BD-Rate improvements but with a lower computational cost. In this work, they developed a mapping for the Quantiser Parameter for H264 to VP9 and AV1. They were able to predict the bitrate and quality of a video compressed by a newer but slower encoder by using features from encoding using a faster, but older, encoder. This cross codec approach shows that there is potential in utilising information learnt about videos on a per clip basis across different video codecs.

Recent work by Ling et al. (2020) indicated that the rate distortion performance of a video can be categorised. They have indicated that information about the video clips, in

particular chrominance information, may lead to better RD-categorisation and subsequently improved RD-optimisation.

Covell et al. (2016) and Aaron et al. (2015) both recognised that by choosing codec hyper-parameters *per clip* significant gains could be achieved. This is because the statistics of video clips varies greatly over any corpus. The idea of adjusting codec parameters based on the content of the video to be encoded has had a few similar names, per title, per scene, per shot or per clip. In this section, we review a few per clip approaches to video encoding. Overall, it can be seen that a per clip approach is beneficial and we apply this idea to the Rate Distortion Optimisation of a video codec.

2.3 Per Clip Lagrangian Multiplier Optimisation

There exists a limited amount of work on adaptation of λ in the rate distortion equation. In all cases, results support the idea that adjusting λ leads to improvement in codec performance *per clip*. The idea is typically to adjust λ away from the codec default by using a constant k as follows:

$$\lambda = k \times \lambda_{\mathrm{o}}, \tag{2.14}$$

where λ_{o} is the default Lagrangian multiplier estimated in the video codec, and λ is the updated Lagrangian.

Zhang and Bull (2019) highlight three research questions which overlap with our research. First is a question of "optimality". Does the current RDO model still provide the optimum rate distortion performance for clips. Next is "independence", which explored the question of whether the optimal Lagrangian multiplier was consistent across various video content. Finally a question of "predictability". If the Lagrangian multiplier is not optimal on a per clip basis, can any features or coding statistics be used to predict the optimum Lagrange multiplier?

2.3.1 Through Classification

One method of determining k is by classifying the content of the video to determine the appropriate adjustment. Ma et al. (2016) proposed an adaptive Lagrangian multiplier algorithm for RDO in HEVC using Machine Learning. Their work used a Support Vector Machine to determine k using a perceptual feature set. That set includes scalars representing Normal Flow, Spatial Information and Temporal Information, and a texture feature based on Gray Level Concurrence Matrix. The spatial perceptual information was derived from the standard deviation of the output from a Sobel filter applied to luminance. The Grey Level Co-occurrence Matrix is a descriptor which combines the contrast, correlation, energy and homogeneity of the video sequence. The Normal Flow is described by the divergence, curl, peakness and orientations within the frames. Their experiments used 37 Dynamic Texture videos (Ghanem and Ahuja, 2010). They reported up to 2dB improvement in PSNR and 0.05 improvement in SSIM at equal bitrates.

Hamza et al. (2019) also take a classification approach but using scene classification into indoor/outdoor/urban/non-urban classes in order to select one of two recommended Lagrangian multipliers. They then used the same k for each class. Their work used the Xiph.org (2018) dataset and reported up to 6% improvement on intra frames of some video frames. This work recognised that the visual information semantics that describe the nature of the image were not considered in the experimental determination of λ for the MPEG based codecs (H263, H264, H265). They do not modify k directly, but apply a factor which would impact the average bits needed across the video sequence. In order to classify the scene, neural networks such as SegNet (Kendall et al., 2015) were used. The value for k is adjusted based on the classification of each macroblock.

2.3.2 Through Regression

As opposed to selecting a value for k from a discrete set, another approach is to determine k through regression based on the features of the video. Yang et al. (2017) used a combination

of features instead of just the MSE ratio ($k = D_P/D_B$) from Zhang and Bull (2019). In their work, they used a perceptual content measurement S to model k with a straight line fit $k = aS - b$. Here again a, b were determined experimentally using a corpus from Xiph.org (2018). The loss in complexity of the fit is compensated for by the increase in complexity of the feature. They report a BD-Rate improvement of up to 6.2%. In their work, similar to Hamza et al. (2019), they highlight that the perceptual features of the video may be of great importance in determining an optimal Lagrangian multiplier.

Zhang and Bull (2019) used a single feature, the ratio between the MSE of P and B frames. This feature gives some idea of temporal complexity. Experiments based on the DynTex database yielded up to 7% improvement in BD-Rate. This work explored using k in the range $0.2 \leq k \leq 5$. Using low resolution (352×288) video clips, they showed that the existing Lagrangian multiplier is not ideal. However, in their process, they were able to determine that the ratio of the distortion of P-frames and B-frames correlated with what they believed to be the optimal λ for a given clip. This work was vital in highlighting the shortcomings of the current system, however was limited in its experiments. It is important that we investigate more than the small dataset presented in Zhang and Bull (2019).

2.3.3 Through Quantiser Manipulation

Since λ is linked to QP as explained previously, another common approach is to adjust λ implicitly through the quantiser parameter, QP. This achieves a similar goal of attempting to improve the RDO of a codec, but has a wider impact as it changes the DCT coefficients in the compressed media as well.

Taking a local, exhaustive approach, Im and Chan (2015) proposed encoding a frame multiple times in a video. Each frame was encoded using a Quantiser Parameter from the set ($QP \in (QP, QP \pm 1, QP \pm 2, QP \pm 3, QP \pm 4)$). They chose the QP which leads to the best bitrate improvement per frame. This led to increased complexity in coding in HEVC, however for a single sequence it achieved a 14.15% BD-Rate improvement.

Papadopoulos et al. (2016) exploited this and applied an offset to QP, in HEVC, based on the ratio of the distortion in the P and B frames. Each QP was updated from the previous Group of Pictures (GOP) using $\text{QP} = a \times (D_P/D_B) - b$ where a, b are constants determined experimentally. This lead to an average BD-Rate improvement of 1.07% on the DynTex dataset, with up to 3% BD-Rate improvement achieved for a single sequence.

2.4 Conclusions

Working, practical codec implementations have eroded the optimality of estimation of the Lagrangian multiplier. That is because the estimation of λ has been performed empirically yielding a number of different fixed relationships between λ for different RDO decisions in codecs. This was inevitable because optimal estimation is just too expensive *per clip*. To make matters worse, historically, experiments were performed on small corpus sizes and on content which does not adequately represent modern media. Hence, it is likely that a better λ exists for an individual video clip which improves the BD-Rate compared to the current defaults used in existing implementations of the codecs considered.

The works described in Section 2.3 have overlooked the possibility that a direct search using BD-Rate as the objective function, could lead to even more improvements. In some way all these previous works have reported data showing k versus $BDRate$, but they use various summary models to simplify this relationship across a corpus. By using instead Direct search optimisation, we are able to explore just how much more BD-Rate gains there are per-clip. This idea is presented next.

Chapter 3

Re-engineering Rate Distortion Optimisation through Direct Optimisation

The previous chapter highlighted the potential of adjusting the Rate-Distortion Optimiser within modern video codecs. This was supported by the review literature which reports improved results in codec performance when selecting an appropriate λ. The idea, summarised here, is to adjust λ away from the codec default estimation λ_o by using a constant multiplying factor, k, across the clip/sequence, as in Eqn 2.14 repeated below.

$$\lambda = k \times \lambda_o$$

While previous works have attempted to create models that can adapt k to the clip content, there has been so far no effort to establish the *best possible* adjustment, k. This chapter will re-establish the need for adjusting λ on a per-clip basis (per-clip optimisation) to gain improvements in BD-Rate. The major contribution here is a method to determine k using direct search optimisation techniques.

In order to show what possible gains can be made by tuning k, we report in Figure 3.1 the BD-Rate improvements obtained at different values of k for two clips (`NewsClip_720P-7745` and `CoverSong_720P-4b12` (from Wang et al. (2019)). We first observe that the optimal value of k can vary substantially between clips. This reinforces the view that the traditional one size fits all approach is sub-optimal.

Secondly, the general shape of the curve is well behaved with no local minimum and just the single global minimum. That minimum represents the maximum BD-Rate improvement available, 2.02% for `NewsClip_720P-7745` and 3.61% for `CoverSong_720P-4b12`. Because of this, numerical optimisation (for minimisation of functions) techniques can be confidently applied to determine the optimal value for k that delivers the best BD-Rate improvement for a given clip. The work presented in this chapter has been published in **"Per Clip Lagrangian Multiplier Optimisation for HEVC."** by Daniel J. Ringis, François Pitié, and Anil Kokaram, in *Electronic Imaging 2020*. Electronic Imaging. January, 2020.

3.1 Using the BD-Rate as the Objective Function

In this work, the objective function for our optimisation is directly chosen to be the BD-Rate (Bjøntegaard, 2001), which is here defined as a function of k, $B_r(k)$ as follows:

$$B_r(k) = \frac{1}{D_2 - D_1} \int_{D_1}^{D_2} (R_{k=1}(D) - R_k(D)) dD, \tag{3.1}$$

where the integral is evaluated over the quality range $[D_1, D_2]$. $R_{k=1}(D), R_k(D)$ are the RD-Curves corresponding to λ_o and $\lambda = k\lambda_o$. Each RD operating point is generated using the same rate control mode (CBR, VBR, CRF) within a range that matches typical streaming media use cases. Then a polynomial-log fit, as recommended in (Bjøntegaard, 2001), is used for evaluating the integral over the entire $[D_1, D_2]$ range. This distortion metric can be PSNR, SSIM or VMAF.

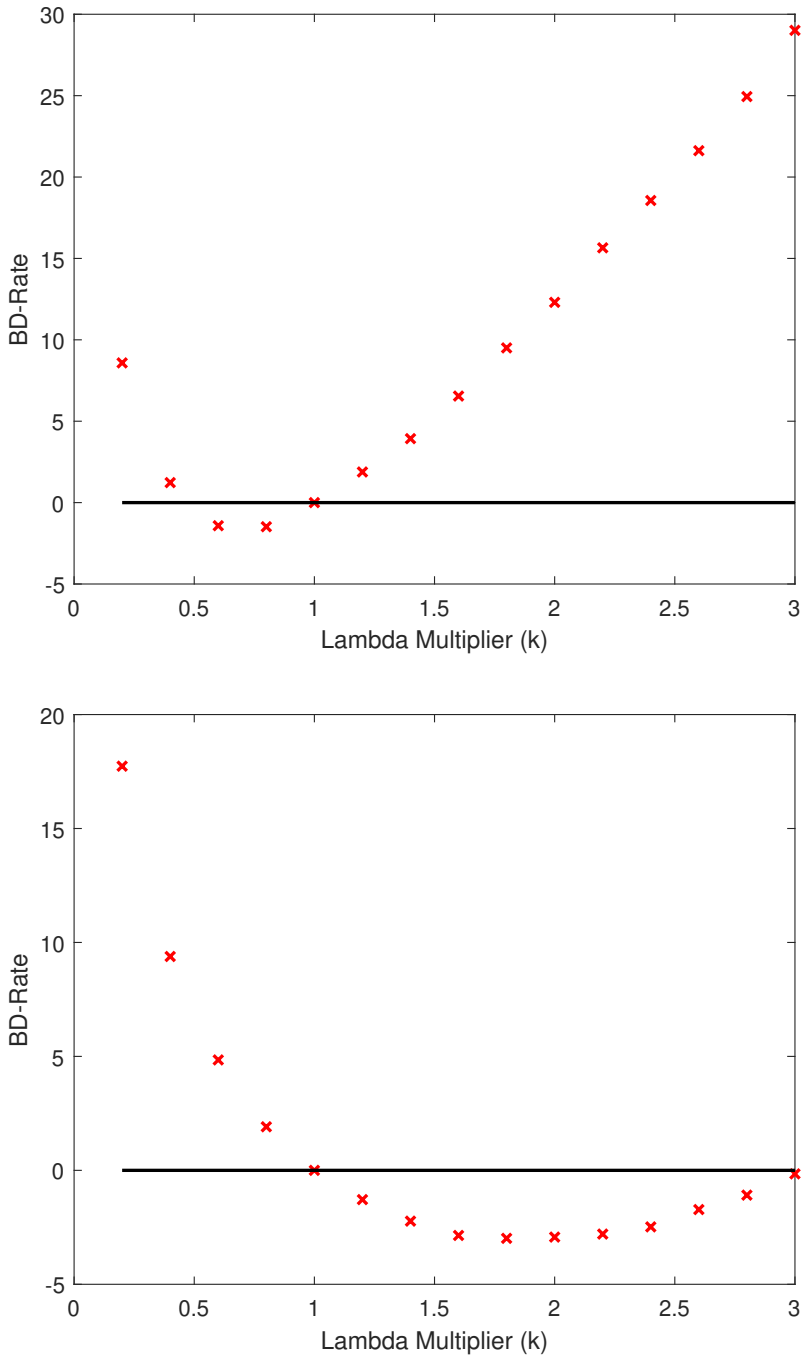

Fig. 3.1 *BD-Rate vs* $k = [0.2 : 0.2 : 3]$ *for two different clips (*`NewsClip_720P-7745` *and* `CoverSong_720P-4b12` *(from Wang et al. (2019)) on HEVC. The best performing BD-Rate gain is achieved at a different value for k in each clip. Therefore, estimating the value of k which provides the best BD-Rate performance for each clip (per-clip optimisation) is sensible. The curve shape shows a global minimum implying that classic optimisation strategies should be successful.*

Note that for every evaluation of $B_r(k)$, a number of encodes, N_e, are required as well as the subsequent BD-Rate calculation itself. This is because each evaluation of $B_r(k)$ requires the generation of a full rate distortion curve. There will therefore be a trade-off between the precision of the RD-Curve and the computational complexity.

In this general strategy, we see that there are a few choices which need to be considered aside from the quality metric. Obviously, the first is what minimisation scheme to use to determine the minimum of k w.r.t. $B_r(k)$. Next, is to determine a reasonable number of operating points, N_e, required to generate each RD-Curve. Finally, which distortion metric to use, and which encoding method to use.

3.1.1 Direct Optimisation Methods

As the shape of BD-R vs k showed that there was no local minimum and just the single global minimum, we can use one dimensional minimisation search methods. For this, three simple direct optimisation algorithms were tested.

Grid Search. In this method, successive refined grid positions in k are searched. The initial grid was $k = 0.2 : 0.4 : 3.0$. The value of k which yielded the minimum $B_r(k)$, k^1_{opt} was calculated based on a spline curve fit to the sparse points, evaluated at steps of 0.01. Successive refinements searching positions of $k^n_{\text{opt}} \pm \Delta$ were then performed with $\Delta = 0.2, 0.1, 0.05$. Figures 3.2 and 3.3 show this process in action using two clips. 'x's indicate the initial evaluated positions, and the solid line is the initial curve fit. The subsequent refinement grids are indicated with +. This process therefore always required fourteen (14) evaluations of the objective function.

Golden Section Search and Brents Method. Traditional methods for finding turning points such as Golden Section Search and Brent's method are well established for finding the local minimum of a unimodal function. These methods are improvements on the simple bisection search (Press et al., 1988). The experiments used the implementations as provided

by `minimize_scalar` function available in `scipy version 1.8.1`. Other minimization solvers could be used in place of Golden Section Search or Brents Method. However, these were selected for their quick off the shelf use (Press et al., 1988).

These are smarter direct search methods using local curve fits. As the Grid search method required fourteen (14) evaluations of the objective function, these optimisation routines a maximum of fourteen iterations was performed with a tolerance of 0.02% in the objective function. This allowed a fair comparison of the strategies deployed.

3.1.2 Illustration of the Problem Using Example Sequences

An example of the search results (k vs BD-Rate) for the methods can be seen in Figures 3.2 and 3.3 for two particular clips. Implementation details are in the next section, but one key observation here is the very different appearance of each curve. This shows that a summary model (e.g. used by Zhang and Bull (2019)) may be unable to exploit all the gains available. Furthermore, it is noted that the points discovered by the direct search methods deviate from the initial smooth curve fit near the minimum. This shows the complexity of the surface in the region of interest. Finally, note that the initial iterations of Brent's and Golden searches cover a very similar curve as the Grid search approach. This means that the complexity of the curves is actually limited to the region near the minimum.

Figures 3.2 and 3.3 (bottom parts) show the points visited in the optimisation search using Golden Section and Brent's method for two other clips (`NewsClip_720P-7745` and `CoverSong_720P-3dca`) from Wang et al. (2019). This shows the optimisation being successful in each case, leading to a BD-Rate improvement of 2.02% (at $k = 0.6$) and 0.61% (at $k = 1.5$) respectively.

These search patterns show that there is a lot of promise in the application of direct optimisation on a per clip basis as opposed to a complete adjustment of the parameters within the codec. In the next sections a much expanded corpus is used to generate a statistically meaningful baseline of performance and then focus on computational complexity.

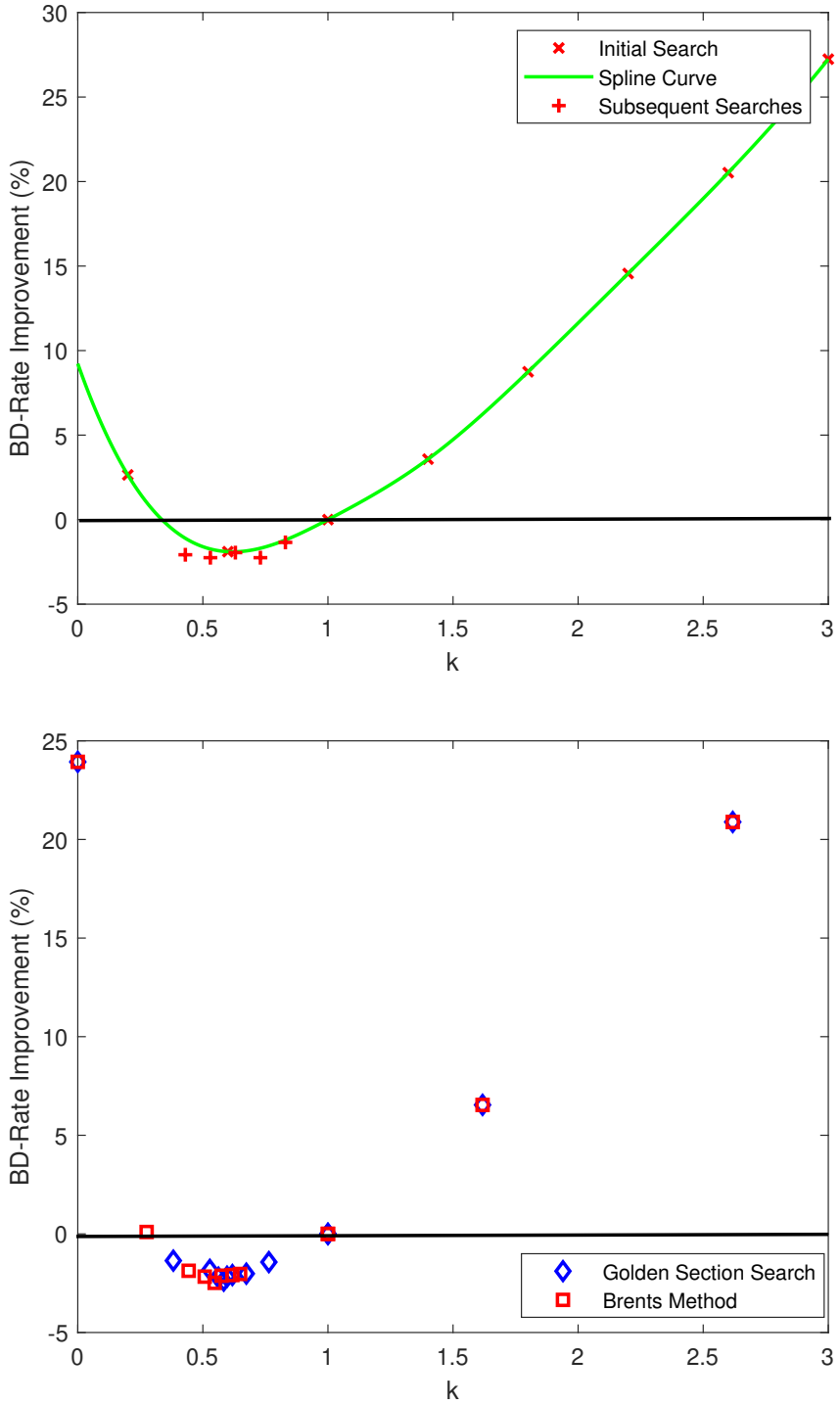

Fig. 3.2 *BD Rate(%) vs k for video* `NewsClip_720P-7745` *from Wang et al. (2019) with HEVC. Top : Multi-Res Grid search, where X's are the initial search results, the solid line is the fit spline interpolation and +'s are the subsequent searches. Bottom : Brents Method* ◊ *and Golden Section* □.

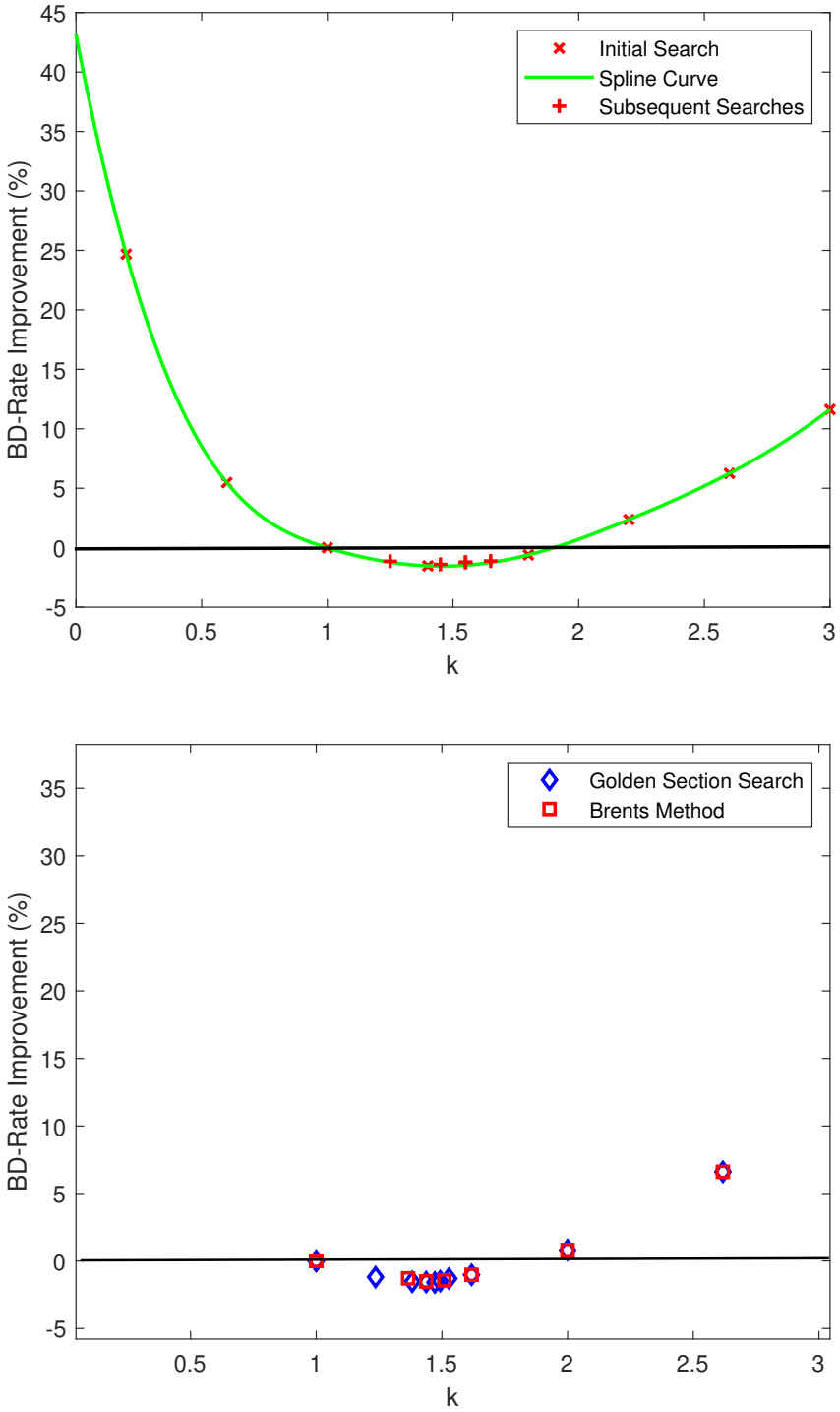

Fig. 3.3 *BD Rate(%) vs k for video* `CoverSong_720P-3dca` *from Wang et al. (2019) with HEVC. Top : Multi-Res Grid search, where X's are the initial search results, the solid line is the fit spline interpolation and +'s are the subsequent searches. Bottom : Brents Method* ◊ *and Golden Section* □.

3.2 Dataset and Experimental Setup

A variety of experiments were run to establish the best possible improvement for two codecs on a per clip basis. There are many decisions required when encoding a video, these include the codec, what is the target distortion metric, the number of RD points in the RD-Curve and the encoding mode. For this work two codecs are used, HEVC (in particular the x265 implementation) and VP9. As mentioned in a previous chapter these were chosen as they were the newest widely available codecs at the time this work begun. The most commonly used metric is PSNR due to its prevalence in the reviewed literature. By using the most common metric, it can be used for quick comparisons to other work as well as an established frame of reference. The other metric used was SSIM. VMAF was considered for these experiments but at the time the preliminary experiments were performed, it was not widely adopted and was unfortunately slower to compute. Future experiments should include VMAF. In order to generate the RD-Curves a couple of decisions need to be made, the number of operating points and the encoding mode. The minimum number of operating points used in comparable literature (Ma et al., 2016; Papadopoulos et al., 2016; Zhang and Bull, 2019) was five, in the range 22:5:42. Two encoding modes were explored, constant bitrate (CBR) and constant rate factor (CRF).

Some preliminary experiments were performed using Constant Quantiser as the target encoding mode, however, there was very little improvement found in our initial testings. We believe that this is due to the inherent link between λ and Q in the implementation of λ in video codecs.

3.2.1 Dataset

Previous work used a small corpus size (approximately 40 clips (Ma et al., 2016; Zhang and Bull, 2019) (up to 300 frames per clip)). Also the types of content used in previous corpora are not necessarily a good representation of modern material. Therefore, in this work,

Table 3.1 *Composition of the dataset. Each segment is 150 frames (5 seconds) chosen to represent a typical DASH streaming segment length.*

Subset	**Segments**	**Resolution**
Youtube UGC (Wang et al., 2019)	8000	320p to 1080p
MCL (Lin et al., 2015)	220	1080p
Dyntex (Ghanem and Ahuja, 2010)	650	240p
Derfs (Xiph.org, 2018)	796	240p to 1080p
Netflix (Xiph.org, 2018)	80	1080p
Total	**9746**	

we use an expanded dataset of 9,746 5-second clips. This includes the recently published YouTube dataset (Wang et al., 2019) representing 12 classes of video as specified by the YouTube team. A sample of each class of the clips can be seen in Figure 3.4. In addition we use clips from other publicly available datasets, these include Netflix dataset (Chimera and El Fuente) (Xiph.org, 2018), DynTex dataset (Ghanem and Ahuja, 2010), MCL (Lin et al., 2015) and Derfs dataset (Xiph.org, 2018). Multiple DASH segments (clips) of 5 seconds (150 frames) were created from each sequence. Table 3.1 shows the database composition by clip categories. There are 9,746 video clips at varying resolutions with a wide range of video content, representative of typical usage. This represents more than a 100 fold increase in the amount of data used for our experiments as compared to previous work. A full list of the segments used can be found in Appendix 2.

By greatly increasing the size of the video corpus used in these experiments, it allows for better representation of the application of encoding for internet video. In particular, datasets like the YouTube-UGC (Wang et al., 2019) and the Netflix (Xiph.org, 2018) dataset directly represent the content two of the most popular video streaming services today. User Generated Content Video was not used in prior research in the area of Rate Distortion Optimisation. With the release of that dataset, our work can potentially highlight the limitations of the existing codec implementations on the most commonly uploaded and viewed style of video (Wang et al., 2019).

Fig. 3.4 *Example frames from each the twelve classes used in the YouTube dataset. Left to right and top to bottom: Animation, CoverSong, Gaming, HowTo, LiveMusic, LyricVideo, MusicVideo, NewsClip, Sports, TelevisionClip, Vlog*

3.2.2 Implementation details

For each codec, the codebase was modified to take k as an argument. Hence this alters the rate distortion constraint from equation 2.14 used in RD control throughout the codec. We use the VideoLAN Organization (2018) implementation of H.265 , `x265`[1] and the WebM Project (2019) implementation of `VP9`[2]. The open source nature of these implementations was the main reason for selecting them as well as encoding speed. With the expectation that nearly ten thousand videos will be encoded multiple times for each run of this experiment, encode time was a priority in the selection of codec implementation.

It is important to note that this work does not aim to compare HEVC and VP9, but to improve both codecs relative to their default behaviour. The command invocation used was as follows:

HEVC: `x265 -input SEQ.y4m -crf <XX> -tune-PSNR -psnr -output OUT.mp4`

VP9: `vpxenc -p 1 -end-usage=vbr -cq-level <XX> -tune=psnr -psnr -o OUT.mkv SEQ.y4m`

where `SEQ` and `OUT` are the filenames for the raw input file and output encoded video. Clips were encoded with `<XX>` in the range 22:5:42.

3.3 Results

Figure 3.5 shows the average BD-Rate improvement for each sequence class for the three direct optimisation methods. For this, we used a subset of seventy-seven YouTube-UGC videos, the list of these videos can be found in Appendix 2. The bar chart shows that in almost all cases, Brent's and Golden methods are better than the multi-res grid search. Brent's method and Golden Section Search outperformed the Multi-Res Grid Search by 0.51% on

[1]Version: 3.0+28-gbc05b8a91
[2]Version: v1.8.1-152-g65c439523

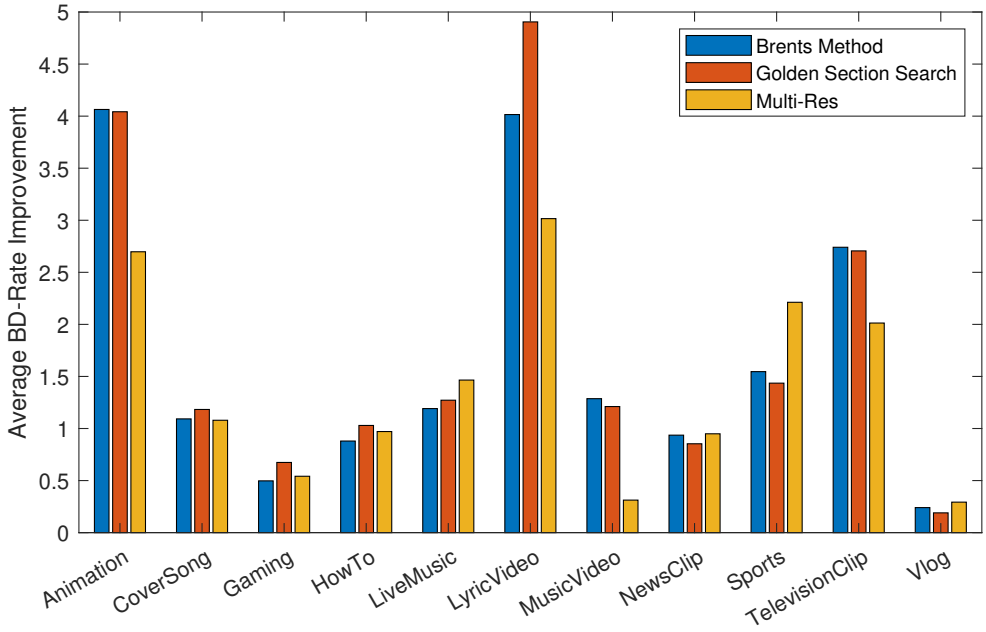

Fig. 3.5 *Average BD-Rate improvement (%) for each class of video. Both Brent's Method and Golden Section Search require less iterations than Multi-Res Grid search and often provide better BD-Rate improvement*

average. This is to be expected as traditional, well established optimisers should outperform the relatively naive, sparse brute force method of our multi-res grid search.

In this small initial experiment we found that our per class gains in BD-Rate are within 0.5%-5%. This is significant as it showed that using a corpus slightly larger than what the established literature used we can get improvements in most of the clips tested. However, for three classes (Live, Sports and Vlog) multi-res grid search provided slightly better results. We believe that this was probably due to the restrictions on maximum allowed iterations and the tolerance thresholds set for the competing methods.

The best BD-Rate improvement for each class is listed in Table 3.2. Up to 21% is observed in an animation clip. This highlights that the gains are bitrate without loss in quality, and that there are significant gains to be had *per clip*. This shows that there are potentially large gains available with per clip optimisation of the Lagrangian Multiplier.

Table 3.2 *Best BD-Rate (%) improvement for each class of sequence using the three direct optimisation methods*

Class	**Brent**	**Golden**	**Grid**
Animation	**21.667**	**21.667**	13.109
CoverSong	1.839	2.279	**2.377**
Gaming	**1.858**	1.762	1.665
HowTo	5.023	**5.239**	5.037
LiveMusic	1.570	**1.810**	1.776
LyricVideo	**15.879**	**15.879**	10.383
MusicVideo	6.417	**6.760**	0.516
NewsClip	**2.481**	2.371	2.251
Sports	**8.391**	7.583	8.378
TelevisionClip	**10.533**	10.238	9.486
Vlog	0.351	0.496	**0.590**

It is very interesting that the Animation and LyricVideo categories provided the best improvements. The majority of frames in these content types exhibit low temporal complexity. That may explain why these categories perform best. Later in this chapter we investigate whether this holds true for the larger corpus.

On average Golden section and Brent's search took 12.6 and 9.7 iterations respectively on this seventy-one clip corpus. This was with a 0.02% BD-Rate improvement tolerance used as the stopping criterion. Given that Brent's method showed comparable BD-Rate improvement to Golden Section Search (Figure 3.5) at a lower number of iterations, we choose to focus on using Brent's optimiser for further experiments. It is possible that with different stopping criterion that even more improvements can be found, however, it may be unreasonable to attempt even more video encodes per clip in the hope of a slight improvement.

3.3.1 Reducing the Number of Operating Points and Optimiser Iterations

In order to see how much potential gain was being "left on the table" by applying the selected stopping criterion, an experiment was run without it. Videos which were encoded without the stopping criterion on the direct optimiser indeed showed additional improvement in BD-Rate. While a further increase in compression performance is welcome, it comes with a heavy computational cost. Figure 3.6 shows an example of convergence in the direct optimisers in this case without the stringent stopping criterion. As can be seen the BD-Rate gains over 8% after 35 iterations. But the lower plot shows the amount of gains achieved after each iteration as a fraction of the final converged rate. Roughly 80% of the potential maximum BD-Rate improvement can be achieved within the first fourteen (14) iterations, therefore the stopping criterion are not that detrimental to performance especially in the light of computational load.

To assess the impact of the accuracy of the RD-Curve, we used more operating points. For this experiment we use <XX> in the range 22:2:42, leading to 11 operating points as opposed to 5. By doing this, we effectively double our computation time. Table 3.3 shows a sample of the BD-Rate improvement found for five clips. In this we can see that there is not a guaranteed improvement in compression performance using the increased number of operating points. In fact the BD-Rate was almost the same on average (0.09% difference in BD-Rate). Because of this, all following direct optimisation experiments used five operating points <XX> in the range 22:5:42.

3.3.2 Per Clip Direct Optimisation Across Our Corpus

From these preliminary experiments it is notable that the worst case scenario for using Direct Optimisation to determine the BD-Rate improvement through adjusting the Lagrangian Multiplier is having the minimum be at $k = 1$, which would lead to a BD-Rate improvement of zero. This means, that it is not possible to get worse through this implementation, but

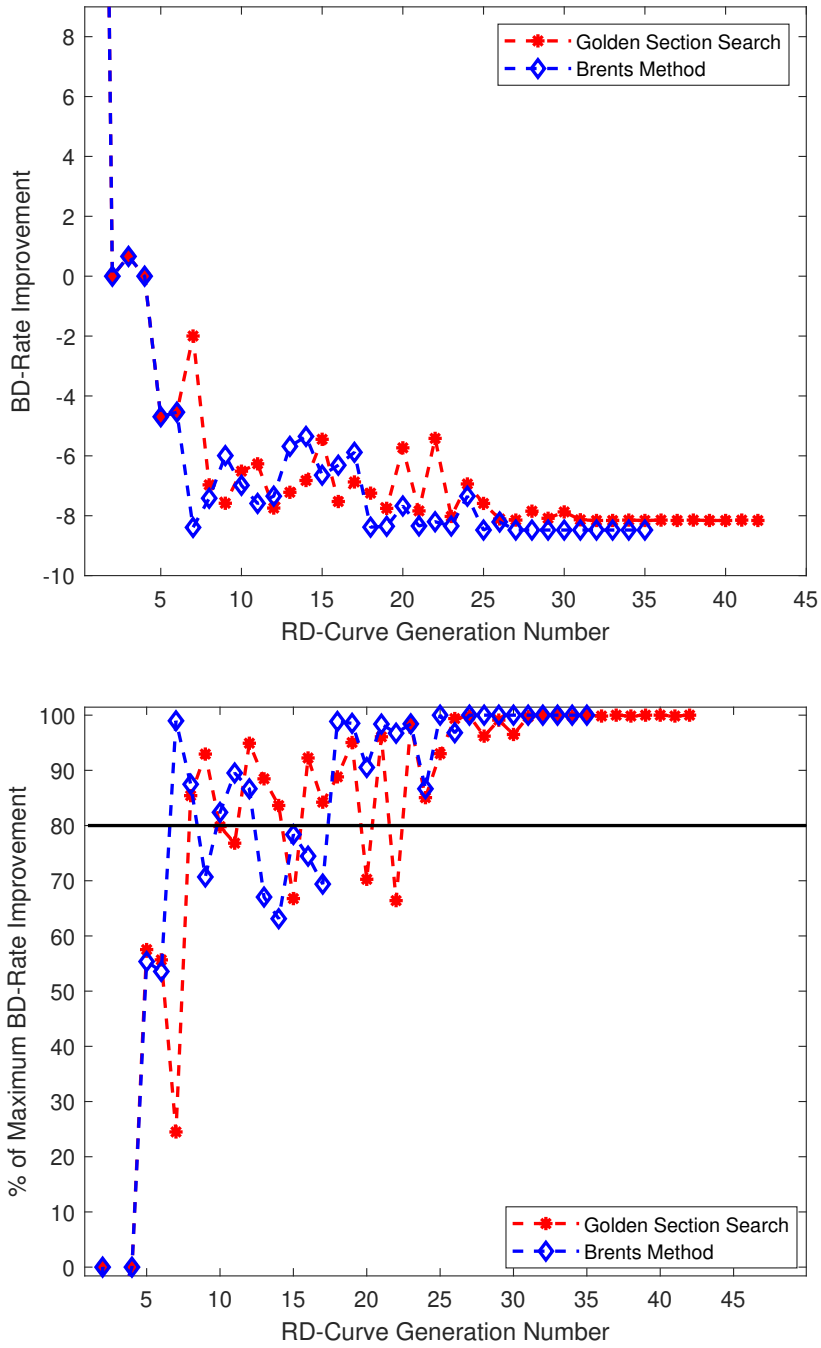

Fig. 3.6 *These two plots illustrate the convergence of the optimisation methods used for a particular clip (Sports_720P-0b9e from Wang et al. (2019)). Top : BD-Rate. Bottom : Normalised BD-Rate improvement. The normalised plot shows the BD-Rate improvement as a % of the maximum BD-Rate possible for this clip. Hence it shows that 80% of the possible BD-Rate improvement is attained within the first 15 iterations.*

Table 3.3 *BD-Rate improvement for five clips using five operating points and eleven operating points. We see that there is no significant difference between the BD-Rate improvement achieved using 5 or 11 operating points. There is an average 0.09% difference in BD-Rate.*

Clip	**BD-R % w/ 5 Points**	**BD-R % w/ 11 Points**	**Difference**
Gaming_720P-0fdb	13.6684	13.4213	0.2471
Animation_720P-31c9	5.70925	5.2514	0.4578
Sports_720P-07d0	2.15307	2.3451	-0.192
MusicVideo_720P-3285	0.69397	0.67234	0.0216
Vlog_720P-3e9c	0.58738	0.62345	-0.0361

only improve or stay the same. By applying this method to the eight thousand clips in the YouTube-UGC corpus we can gain a better understanding of the impact of direct optimisation. We can see the BD-Rate improvement by using Brent's method of direct optimisation for each clip in the YouTube-UGC corpus in Figure 3.7.

The results show that the majority of clips have small improvements, with a few outliers which have considerable BD-Rate improvements. It is interesting to note that these best performers have some similarity in video content, being mostly simple videos with lower temporal complexity. Examples of the image sequences can be found in Figures 3.8, 3.9 and 3.10.

It is noticeable that these are relatively static images, with very little or no motion in majority of the frames. This matches the early observations made about videos in the Animation and LyricVideo clips outperforming the other categories. Low motion images lead to high performance.

While it is useful to see the improvement by each individual clip in the corpus, it is better to quantify the impact of these experiments with a summary of the gains found. In order to best represent the improvements made across our corpus, Figure 3.11 shows the fraction of the clips encoded with our system yield an improvement of at least X%.

The result validates the usefulness of direct optimisation. The best BD-Rate improvement here is 23.86%, 27.05% (HEVC, VP9). 50% of the clips in the corpus show a BD-Rate

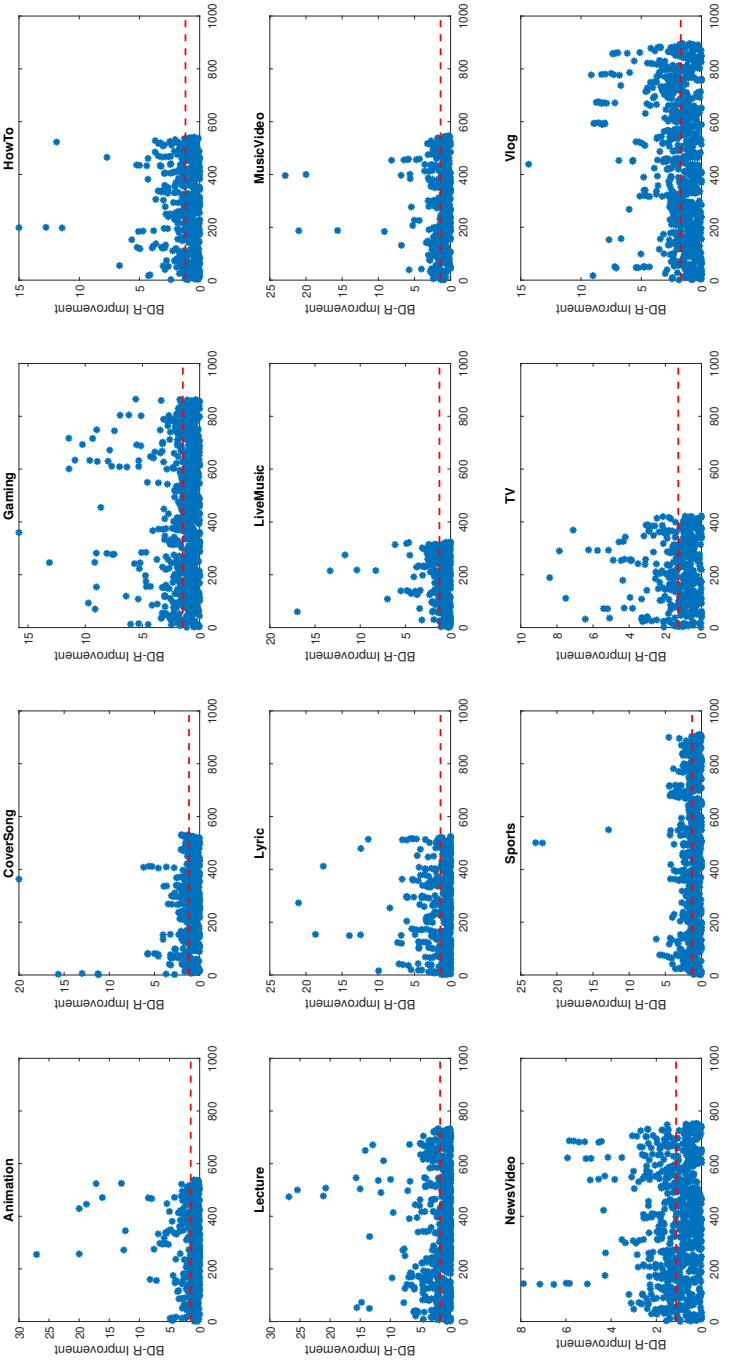

Fig. 3.7 *BD-Rate improvement for each clip in the YouTube-UGC corpus. The dashed red line represents the average BD-Rate improvement for that class of video. In these plots, the higher the improvement, the better the performance. We see that there are a few outliers which have excellent improvement. However, the corpus on average shows 1-2% improvement*

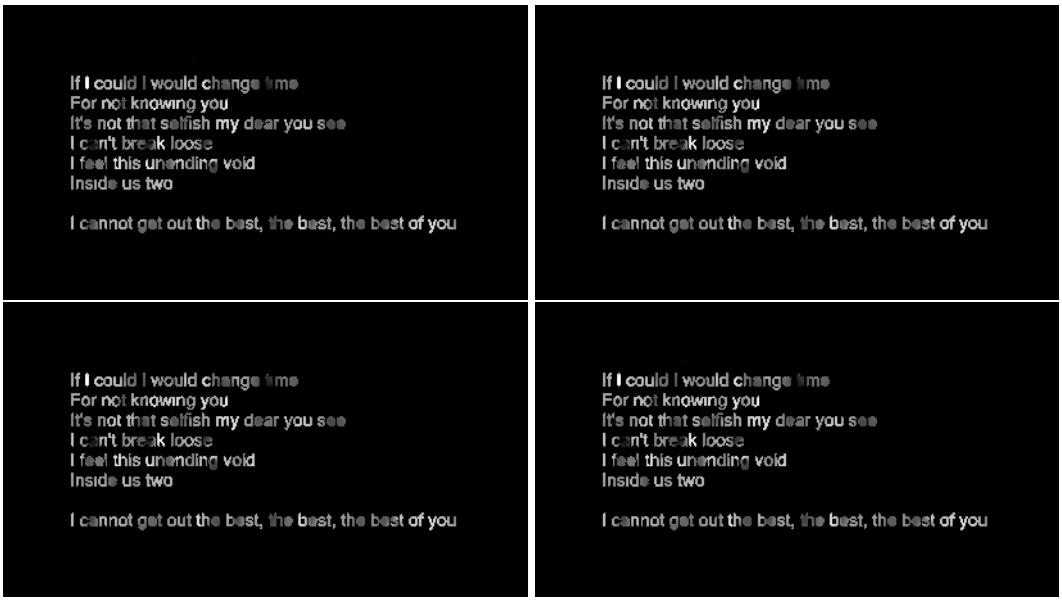

Fig. 3.8 *First frame from each second of one of the best performing clips in the video corpus: LyricVideo_720P-2d24 from Wang et al. (2019). This clip had greater than 15% improvement using the direct optimiser and are very static images.*

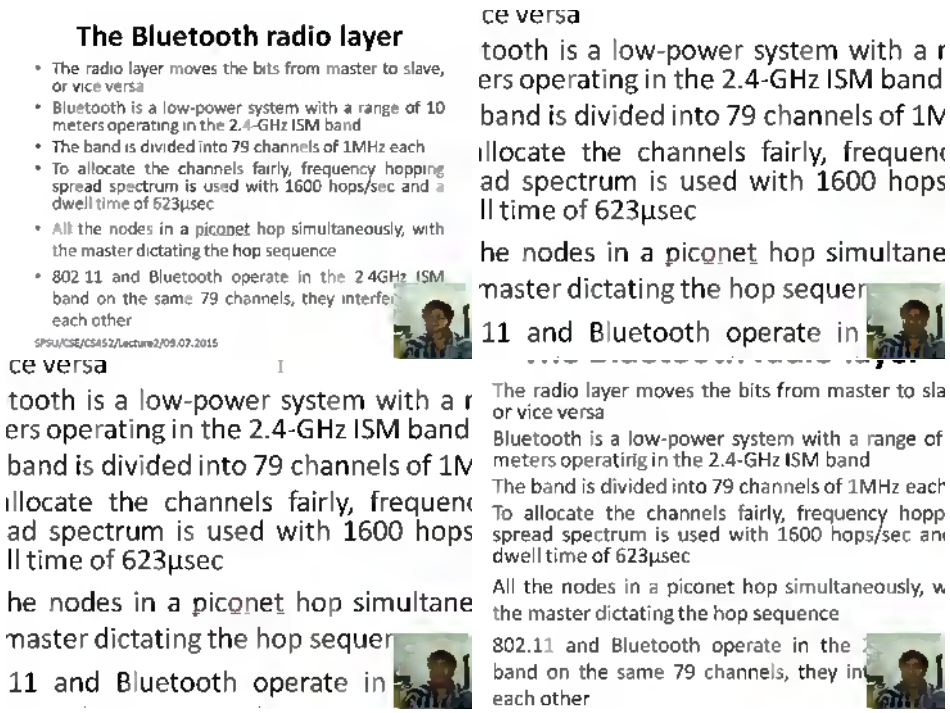

Fig. 3.9 *First frame from each second of one of the best performing clips in the video corpus: Lecture_480P-7e55 from Wang et al. (2019). This clip had greater than 15% improvement using the direct optimiser and are very static images.*

Fig. 3.10 *First frame from each second of one of the best performing clips in the video corpus: Gaming_480P-6d1e from Wang et al. (2019). This clip had greater than 15% improvement using the direct optimiser and are very static images.*

improvement of 0.92% for HEVC and 0.40% for VP9. We can also see that 47% of our corpus in HEVC and 25% in VP9 show a $> 1\%$ BD-Rate improvement.420

We can also see these results broken down by video class as seen in Figure 3.12 as well as zoomed views provided by Figure 3.13. The classes which have the most to gain from our system are the Lectures, Lyric Videos, Vlogs. Those which have the least to gain overall are Television, Gaming and Sports. This indicates that there is a potential link between the content of the video and the BD-Rate improvement using direct optimisation.

Of course there are still a number of video clips where the original Lagrangian multiplier was the best (8% for HEVC, 22% for VP9). Across our large corpus, the direct optimiser took on average 11.7 iterations. Given five operating points for each RD-Curve evaluation (iteration), this implies total of $5 \times 11.7 = 60$ video encodes per clip. This is manageable with low-resolution videos (eg 144p) which encode quickly, but unwieldy at a more typical resolution (eg 720p, 1080p).

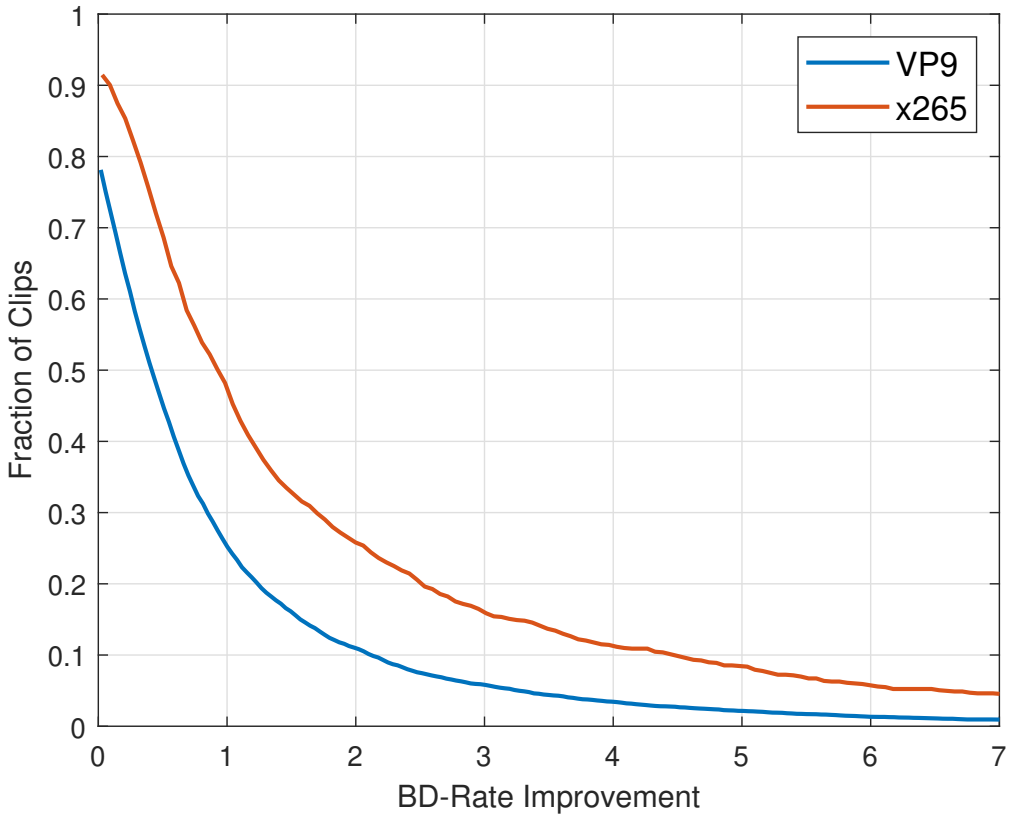

Fig. 3.11 *CDF plots for BD-Rate improvement for the YouTube-UGC corpus. Ideally, we would like these graphs to show all of the corpus having significant improvement. The best BD-Rate improvement here is 23.86%, 27.05% (HEVC, VP9), which extends beyond the limits of the figure. We can see from this plot that 50% of the clips in the corpus would have a BD-Rate improvement of 0.92% for HEVC and 0.40% for VP9. We can also see that 47% of our corpus in HEVC and 25% in VP9 show a* $> 1\%$ *BD-Rate improvement.*

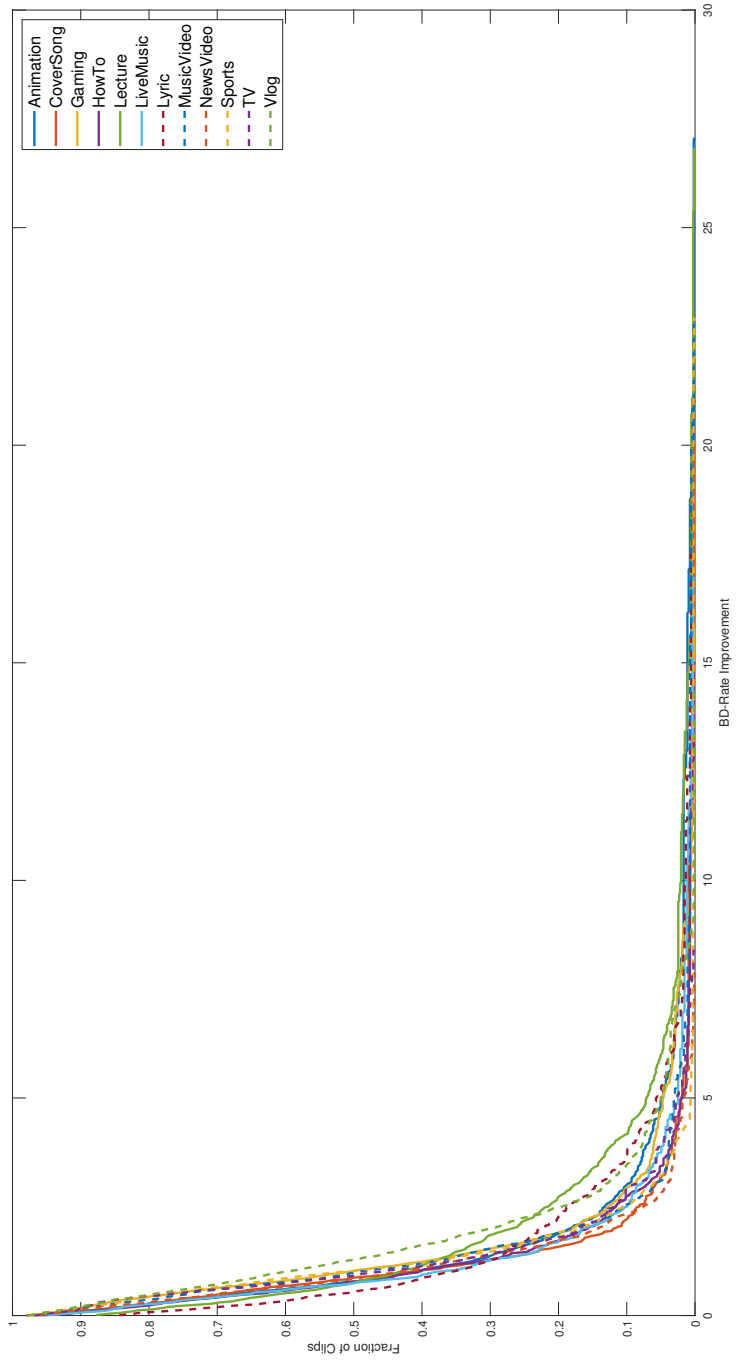

Fig. 3.12 *CDF plots for BD-Rate improvement for the YouTube-UGC corpus for each of the established classes of video using HEVC*

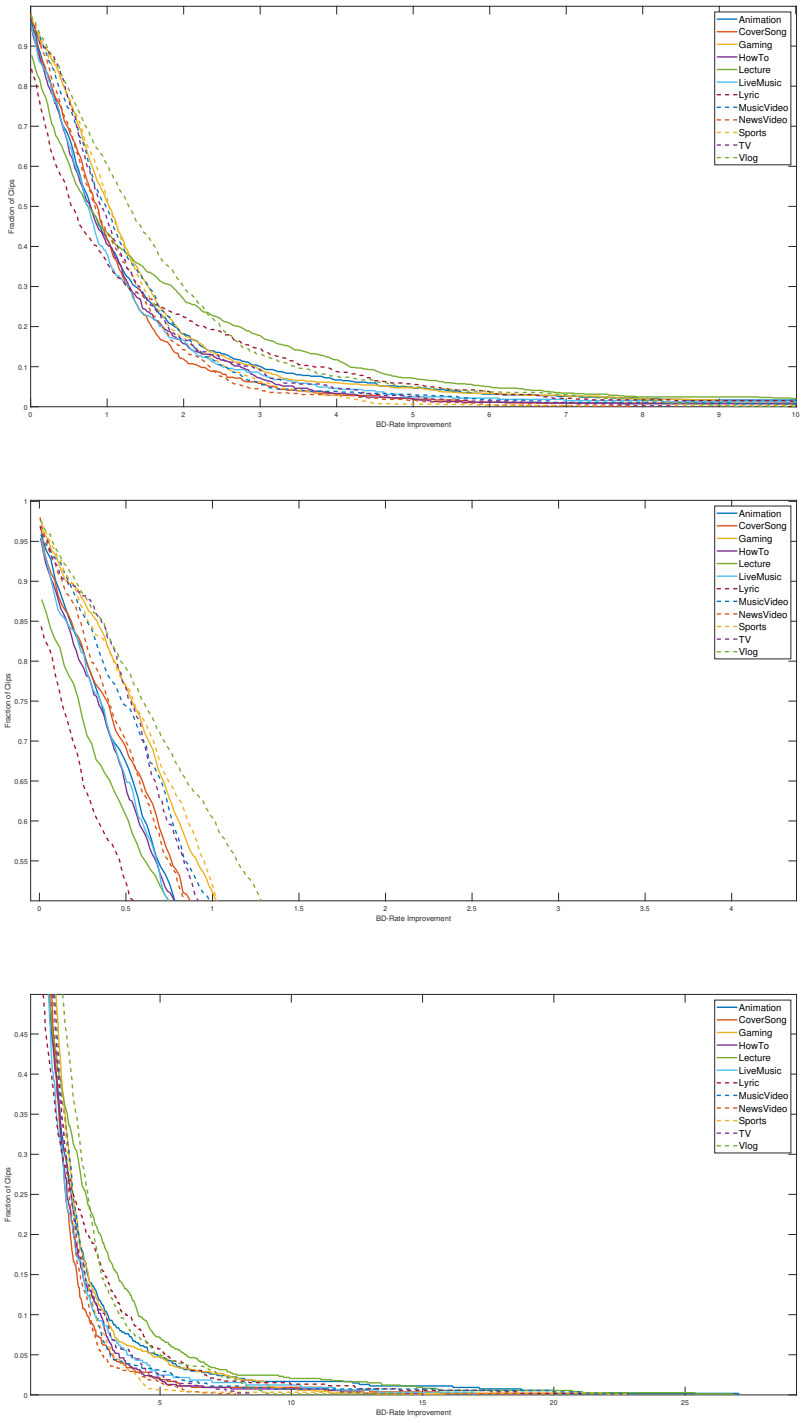

Fig. 3.13 *Zoomed views of the CDF plots for BD-Rate improvement for each of the established classes of video in the YouTube-UGC corpus seen in Figure 3.12. Top is majority of clips, which have an improvement of 0%-10%. We see that overall, Lectures and Vlogs had the best improvements. Middle is the bottom 50% of compression improvement, we see that majority of the Television, Gaming and Sports clips are in this region. Bottom we have the best performers, which continue to be the Vlogs and Lectures as well as Lyric Videos*

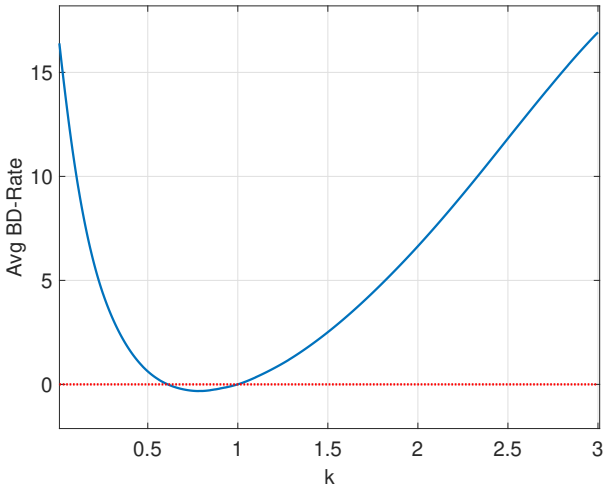

Fig. 3.14 *Average BD-Rate for a given k for HEVC using the entire corpus. This experiment was performed by encoding videos using k=0:0.001:3, and calculating the BD-Rate. We can see that values of k in the range* $[0.7, 0.8]$ *give a clear average improvement (negative BD-rate) across the entire dataset. This may indicate that the adjustment to the Lagrangian Multiplier should be a constant throughout, however further investigation showed otherwise.*

3.3.3 Global Direct Optimisation Across Our Corpus

Another experiment to assess whether the per clip approach is useful at all is to estimate the best performing value of k across the corpus and apply that to all videos. We calculate the BD-Rate over our corpus for values of k in the range k=0:0.001:3. The average BD-Rate improvement for a given k can be seen in Figure 3.14. The figure shows an improvement where k is approximately in the range 0.7:0.8, with a best average improvement coming at $k = 0.782$. This also matches with the histogram of the optimal k values found with direct optimisation on the corpus videos (see Figure 3.15). The following command invocation was used for all video encodes:

```
x265 -input SEQ.y4m -crf <XX> -tune-PSNR -psnr -output OUT.mp4
```

where `SEQ` and `OUT` are the filenames for the raw input file and output encoded video. Clips were encoded with `<XX>` in the range 22:5:42.

For $k = 0.782$, about two-thirds of the clips have an improvement. Overall, we get an average gain of 0.32% across our corpus. This indicates that the default Lagrangian Multiplier

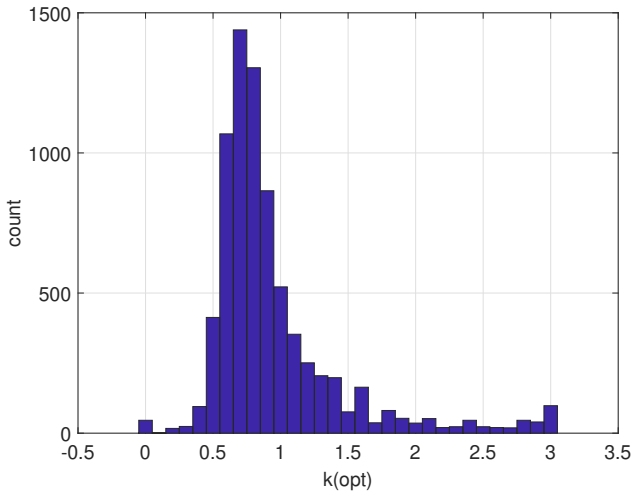

Fig. 3.15 *Histogram of optimum k (and by extension optimum Lagrangian multiplier) determined by Direct Optimisation using Brent's Method for HEVC. As expected from the experiments using k=0:0.001:3 the optimum is most frequently found in the k=0.7-0.8 range, but there is still a wide range of k which has a positive impact on compression performance.*

for HEVC may not be the best. Hence for HEVC, we should adjust the Lagrangian Multiplier to be $0.782\times$ its current value.

Figure 3.16 shows the BD-Rate improvement by using direct optimization of the Lagrangian Multiplier across each clip in the corpus. In this graph, we see the BD-Rate improvement on the x-axis and the fraction of the dataset which was able to achieve that BD-Rate improvement or better. Ideally, we would want this curve to be as close to the top right as possible, as that would indicate all clips achieved high BD-Rate improvements. The key result of this work is that 95% of the clips showed some BD-Rate improvement compared to the default Lagrangian Multiplier, with 46% of them showing a BD-Rate improvement of 1% or better.

On that plot, the CDF corresponding to $k = 0.7$, 0.8 or 0.782 fixed over the corpus is shown. All of those CDFs are worse than the per clip approach, e.g. only 66% of clips show any improvements compared to 95% with per clip. This confirms what we initially hypothesised that a *per clip* approach is best suited for improving compression across the large corpus of video.

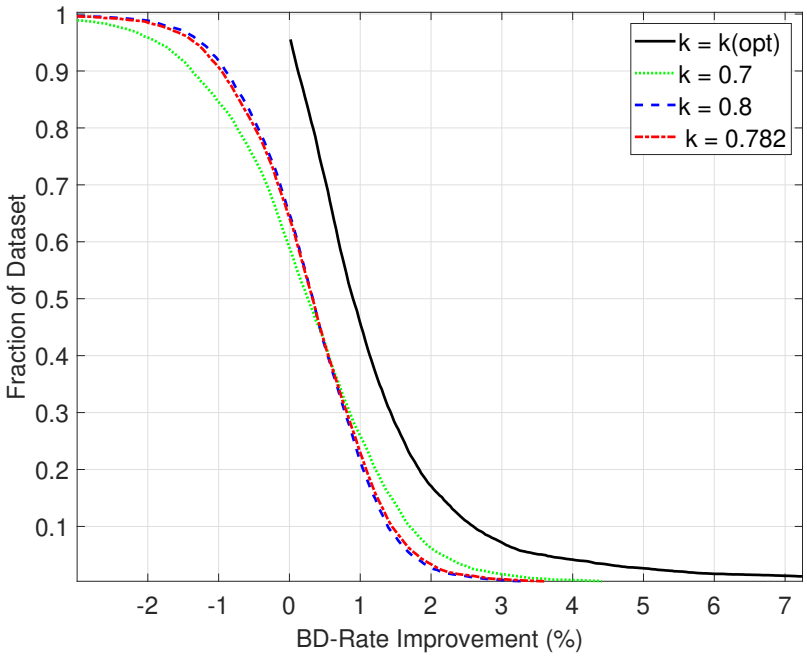

Fig. 3.16 *BD-Rate improvement vs fraction of dataset which achieves that improvement or better through direct optimisation of the Lagrangian Multiplier using Brent's Method in HEVC. We can see from this curve that the per clip direct optimization method (k_{opt}) was capable of achieving BD-Rate improvement in 95% of the clips in the corpus. We can also read off of this graph what fraction of the clips had a 1% or better BD-Rate improvement (47%) and what fraction of the clips had a 5% or better improvement (3%). Also shown are the CDF plots when using fixed k for the whole corpus at k = 0.7, 0.8 or 0.782. Ideally, we would want these plots to be as close as possible to the top right and coincide with the optimal per-clip plot (in black). We see for k = 0.7, 0.8 or 0.782 positive BD-Rate improvements for 60-70% of the clips but worse performance for 30-40% of clips.*

3.3.4 Comparisons to State of the Art

Table 3.4 compares our per clip direct optimisation approach to the results from Zhang and Bull (2019). We examine the BD-Rate improvement in nine clips. These clips can be classified as live action video Akiyo, News, Silent, Bus, Tempete, Soccer available from Xiph.org (2018) and DynTex sequences Shadow, Shower, Wheat from (Ghanem and Ahuja, 2010). We see that the BD-Rate improvement found in the six natural sequences was better for our system. However, the improvement seen in the Dyntex clips was worse than the results reported in Zhang and Bull (2019).

We also re-implement the work presented by Zhang and Bull (2019) to apply that work to modern video. The comparison is not strictly fair because we do not adapt the k every GOP and instead use a single GOP for all the clips. In Table 3.5 we present the BD-Rate and k for eleven clips from Wang et al. (2019) which compares our per clip direct optimisation to our implementation of Zhang and Bull (2019). As can be seen our proposed algorithm estimates values of k that are quite different, and subsequently the BD-Rate improvement is much better than that using k as detailed in Zhang and Bull (2019).

The prior work focused on the creation of models using relatively small corpora to establish an improved λ *per clip*. The size of the corpus used for their experiments were quite small and we know that average performance is worse than individual adaptation as discussed previously. Unlike methods discussed in Papadopoulos et al. (2016); Zhang and Bull (2019) which updates λ and QP per frame, our work selects a single optimal value of λ for the entire clip. This may be a limitation of this work and adjusting λ for an entire clip as opposed to per GOP or per frame may lead to further improvements in the future.

3.4 Conclusions

In this chapter we introduced a strategy for employing "off-the-shelf" optimisation techniques, such as Brent's method, for Lagrange multiplier estimation in a codec (Press et al., 1988).

Table 3.4 *Selected Clips showing BD-Rate improvement using our direct optimisation (BD-R(k_{opt})) as well as reported results from another adaptive Lagrangian Multiplier(BD-R(Zhang and Bull, 2019)) We see that our system is better applied to natural content of live action video Akiyo, News, Silent, Bus, Tempete, Soccer available from Xiph.org (2018) with larger BD-Rate improvement in all six clips. Unfortunately it is not as successful on the DynTex sequences Shadow, Shower, Wheat from (Ghanem and Ahuja, 2010) as the BD-Rate improvement reported in these three clips is larger. We believe that this is due to the nature of the DynTex clips being visually complex sequences (Ghanem and Ahuja, 2010).*

Sequence	**BD-R(k_{opt}) %**	**BD-R% (Zhang and Bull, 2019)**
Akiyo	3.9	2.5
News	2.6	2.6
Silent	1.9	1.4
Bus	0.8	0.7
Tempete	0.89	0.8
Soccer	1.01	1.0
Shadow	0	2.1
Shower	0.62	2.1
Wheat	0.59	2.2

Table 3.5 *Comparison of the BD-Rate improvement using an estimated k from our implementation of Zhang and Bull (2019) vs k determined by our direct search (Brent's method) using clips from Wang et al. (2019). As can be seen our estimated values are quite different and lead to much improved performance. However our adaptation scheme is different from that in Zhang and Bull (2019). For this negative BD-R implies worse performance than the unmodified codec, we see in every clip that our system has a better BD-Rate improvement.*

Clip	**k_{ZB}**	**BD-R (%)**	**k_{Ours}**	**BD-R (%)**
Animation4268	3.173	-5.81	0.777	0.591
CoverSong7360	3.444	-4.23	1.071	0.178
Gaming6403	3.132	-6.18	1.438	0.683
HowTo21c6	2.938	-3.15	0.438	5.239
LiveMusic6452	3.632	-0.85	1.145	0.234
LyricVideo739a	2.491	0.89	1.699	1.412
MusicVideo3285	4.298	-10.21	1.120	0.068
NewsClip7745	2.571	-2.67	0.547	2.482
Sports0b9e	3.108	1.89	1.784	8.391
TVClip02b8	2.112	-3.12	0.587	2.887
Vlog11c5	3.579	-2.12	1.096	0.166

The approach in some sense represents an upper bound for what can be achieved by adjusting the Lagrangian multiplier.

The first major takeaway of this chapter is that there are significant improvements to be found in directly optimising the rate distortion equation on a per clip basis. With almost 30% BD-Rate improvement found on some clips on H.265 or VP9, there is a lot of potential in this work. There can be a great impact in the field of video streaming, if uploaded videos are streamed in the thousands to millions of views range, a bitrate savings of as small as 0.1% can have an impact.

The next major takeaway of this chapter is that optimising video codecs using a corpus which better represents modern video, leads to BD-Rate improvements. By using the video corpus of almost 10,000 video clips, the majority of which are user generated video content, we can find potential positive outliers in improvement. The videos which are fairly static in nature, which are fairly commonly uploaded, such as animation clips, lecture slideshows and lyric videos seem to have significant improvements when the RDO modules within the codec are adjusted.

The results also show that the relationship between k and BD-Rate varies substantially between clips, and that the region near the minimum is not a simple surface, i.e. near the minimum is not an easy area to fit a curve. The direct optimisation done in this chapter however, comes at a high computational cost. Even with a fairly relaxed stopping criterion and using five operating points per rate distortion curve, each video would need to be encoded approximately sixty times (average 11.7 iterations with 5 operating points for each RD-Curve) in order to find the improvement through direct optimisation.

These observations point to further work in *per clip* optimisation using methods which reduce the computational cost. In Chapters 5 and 6 we investigate using proxy videos or more sophisticated machine learning techniques that exploit deeper features in the content to achieve comparable gains from Direct Optimisation at a lower cost.

Chapter 4

Per Operating Point Rate Distortion Optimisation

In the previous chapter we introduced a strategy for employing “off-the-shelf” optimisation techniques such as Brent’s method for Lagrangian multiplier estimation in a codec (Press et al., 1988). In this approach, referred to as our *direct* method, we focused on improving the BD-Rate, corresponding to the best improvement across a series of operating points. In practice, content delivery systems focus on a few specific target bitrates, as opposed to the entire range because video content delivery systems are only required to deliver to a few target formats.

This approach of optimizing across the entire bitrate range of the RD curve is certainly beneficial, but it is potentially sub-optimal. This is because the Lagrangian multiplier has a different impact at different operating points, and there may be additional gains to be found by directly optimizing the Lagrangian Multiplier in smaller ranges of the RD-Curve. This will come at the cost of increased number of video encodes, but a clearer picture of what is the upper bound of bitrate savings using this per clip approach. In this Chapter, we investigate the potential improvements by adjusting the Lagrangian Multiplier within codecs on a per bitrate level. The work presented in this chapter was published in **"Per-clip and per-bitrate**

adaptation of the Lagrangian multiplier in video coding." by Daniel J. Ringis, François Pitié, and Anil Kokaram, in A*pplications of Digital Image Processing XLIV,* vol. 11842, pp. 185-194. SPIE, 2021.

4.1 Introductory Example

To better understand the problem at hand, we can consider a single clip scenario. Using the clip Sports_720P-5bfd (Figure 4.1 from (Wang et al., 2019)), we can see how this approach is sub-optimal by observing the best performing k in three sections of the Rate Distortion Curve.To do this, we can generate multiple RD-curves with k in the range of 0.1:0:1:3.0. We can see the overall result in Figure 4.2, in which we show that k=1 is not the best performing clip. This is to be expected as established in the previous chapter where the majority of clips respond positively to adjustments to the Lagrangian Multiplier. If we look at different regions of the Rate Distortion Curve, low bitrates, middle bitrates or high bitrates, as seen in Figure 4.3, a different value of k provides the best possible improvement in each range. If we combine the different optimal values of k in each bitrate rage a 1.236% BD-Rate improvement results for this clip, whereas the direct optimisation method provided us with 1.143% BD-Rate improvement. Hence gains that we reported so far are not optimal for each individual point.

Because the practical scenario of video streaming would typically deliver at a single operating point, as well as the evidence that the Lagrangian Multiplier adjustment is not ideal at every operating point within a RD curve, we should look to determine what is the optimal adjustment at each bitrate within the operating range.

4.2 Pareto Optimal

In theory, our aim is this:

$$D_{\text{Pareto}}(R) = \sup_k(D(R,k)) \tag{4.1}$$

Fig. 4.1 *Frames from the sequence* `Sports_720P-5bfd` *from (Wang et al., 2019). This sequence had a 1.143% BD-Rate improvement using the Direct Optimisation methods detailed in Chapter 3 on the HEVC codec*

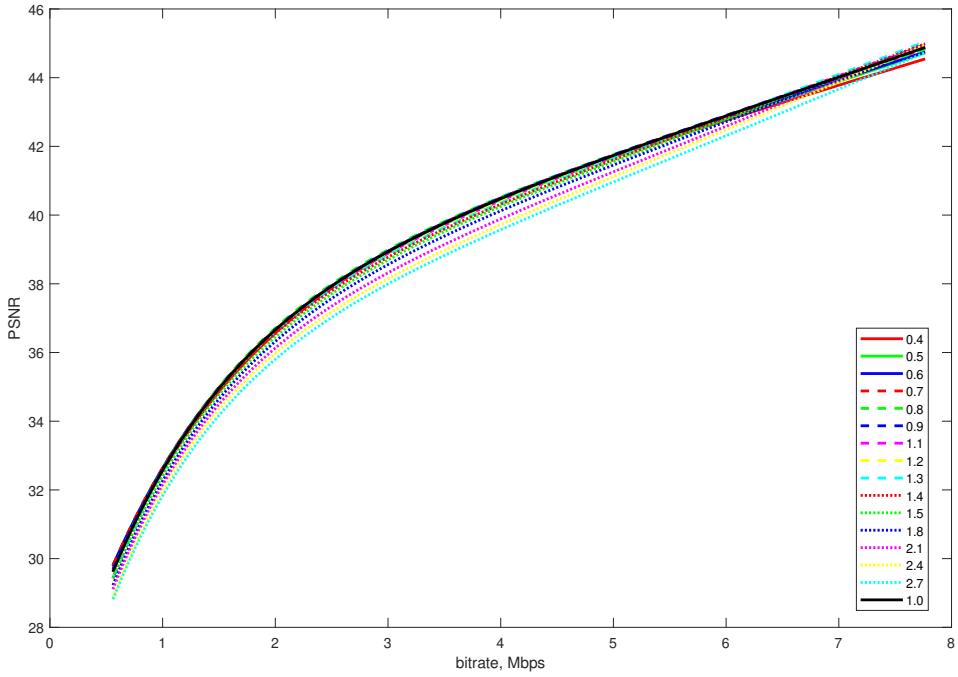

Fig. 4.2 *Using k in the range 0:1:0.1:3.0 to compress a single clip, Sports_720P-5bfd, with constant bitrate encoding in HEVC. Each curve is a different value of k. We can see that for this clip, that k=1 is not the best, which is common for clips in our corpus but the best value for k changes based on the region of the rate distortion curve. Another way to interpret this is that the RD-curves corresponding to different k cross each other at different bitrates*

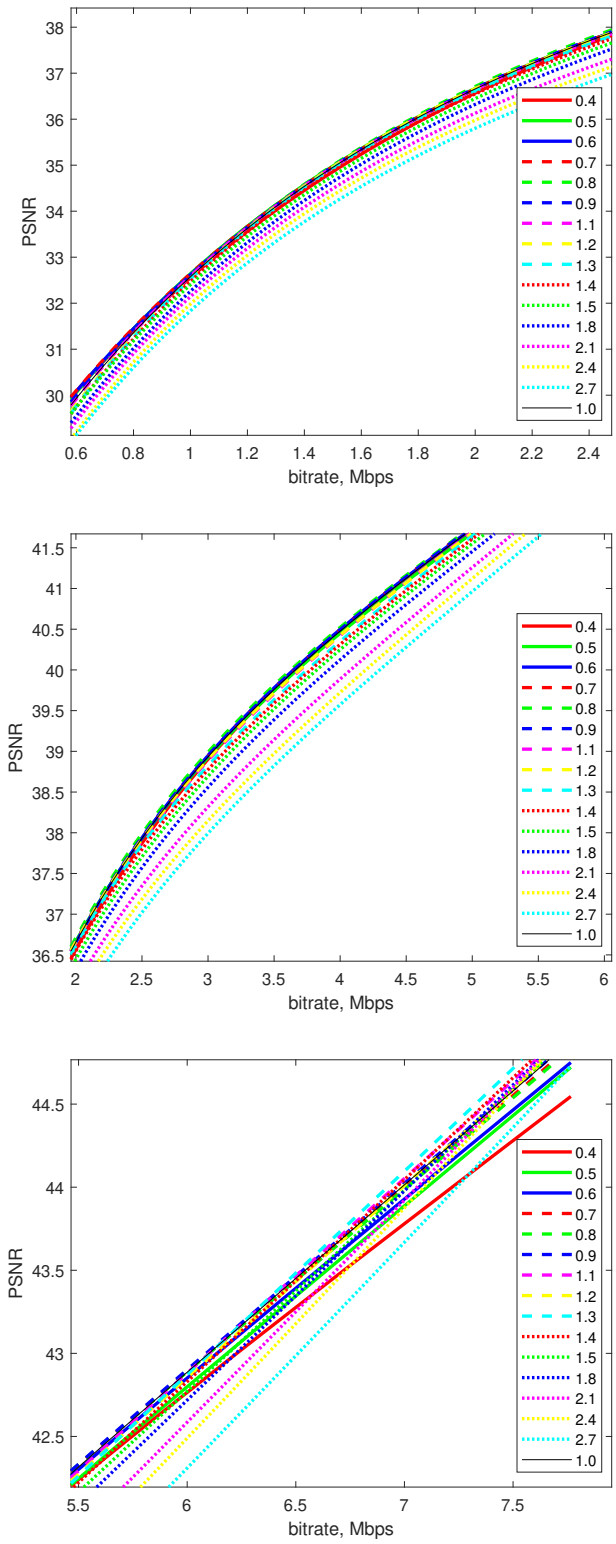

Fig. 4.3 *Zoomed views of Figure 4.2, where a single clip was encoded using HEVC, constant bitrate encoding. Top, is the low bitrates, in this we can see that the best performing value of k is 0.4 in the bitrate range of <2.5Mbps. In the middle, a bitrate range >2.5Mbps and <6.5Mbps,* $k = 0.8$ *is the best performer. Bottom image shows the high bitrates(>6.5Mbps)in which the best performer is* $k = 1.3$*. This indicates that applying an adjustment to the Lagrangian Multiplier for an entire RD-Curve may be limiting the BD-Rate improvement we can achieve.*

however, in practice only a sparse sampling of $\{D(R_i,k_j)\}_{i,j}$ is available from our experiments.

Using the same interpolation scheme as used in the construction of the RD-curve (Bjøntegaard, 2001), we can estimate, for a particular k_j, an approximation $\tilde{D}(R,k_j)$ of D for any bitrate value R. This solves our sparse sampling problem. We have some confidence in the approximation $\tilde{D}(R,k_j)$ because the shape of the curve is well established to be approximately exponential (Bjøntegaard, 2001).

So our estimation becomes

$$D_{\text{Pareto}}(R) = \max_j(\tilde{D}(R,k_j)) \tag{4.2}$$

However, taking the maximum for k_j requires us to sample values for k that are indeed close to the optimum. Interestingly, the direct optimization that we proposed allows to focus the sampling of k_i in the area of interest. However the previous chapter focuses on optimizing the BD rate improvement across the full range of bitrates for a given video clip.

Essentially, what we are looking for is the Pareto optimal of the set of all lambda values used when encoding a given clip at a particular bitrate. That curve is made of the supremum of all these curves.

To obtain this Pareto curve, we would need to find the optimal value of k for any given bitrate while not reducing the quality at this bitrate. We take an empirical approach i.e. we generate a large amount of possible pairs $(bitrate,k)$ and sample the Pareto surface directly from those points. As exhaustively encoding video at each bitrate with a large range of k is not feasible, we need to establish a reasonable process to extract our pairs from our measurements in order to get our Pareto-optimal curve.

In this work, we need to sample k in such a way that it optimizes BD-Rate w.r.t. bitrate. That would in detail lead to many invocations of encoding, so we propose here to define three bitrate ranges as detailed in the Table 4.1. The expectation is that the optimal value of k for any given bitrate will then fall close enough to one of the three sets of values generated

by our three experiments. In detail therefore we expand the range of candidates ($R,\ k_i$) by running 3 direct optimizations at low, medium and high bitrate ranges as specified in that table.

This is not the exhaustive case to determine the best possible bitrate improvement, so there will be even more gains to be achieved if one were to apply our direct optimisation methods at each bitrate used for a given video. This will be a ridiculous amount of computation performed for what may be very little improvement. It was decided that approximating would be the better approach. As the operating points of these curves are discrete, we need to determine what would be the expected quality/distortion at a given bitrate for a given k for each curve. By interpolating, we ensure that we do not have to do an exhaustive number of video encodes. We use exponential interpolation to determine the expected distortion at bitrates in between operating points. For each bitrate we take the point with the best quality as our best performer and combine them. We believe this to be our theoretical upper bound of bitrate improvement for each clip. While there is no guarantee that this would be optimal, because we take values which come from the optimal BD-Rate in smaller ranges , it is reasonable to believe that the points on this would be optimal. Also, as RD-Curves are generally very smooth, we expect that interpolation over a small range is likely to give us a correct value. We present implementation details and results next.

4.3 Experiments

In order to strike a balance between finding the best performance across multiple bitrates using the Pareto-Optimal system and reasonable computational expense, we compute the RD-curves for our sequences at discrete points and then fit curves to estimate the expected performance at the bitrates in between our selected points. We use a subset of the corpus of video described in the previous chapter. We are using the HD (720p and 1080p) clips from the recently published YouTube dataset, (Wang et al., 2019),representing 12 classes of video.

Table 4.1 Operating points selected for the LOW, MED and HIGH ranges

	CRF	**CBR**
LOW	22:2:32	256kbps, 512kbps, 1Mbps, 2Mbps, 4Mbps
MED	27:2:37	1Mbps, 2Mbps, 4Mbps, 6Mbps, 8Mbps
HIGH	32:2:42	4Mbps, 6Mbps, 8Mbps, 10Mbps, 12Mbps

Multiple DASH segments (clips) of 5 seconds (150 frames) were created from each sequence giving us an approximately two thousand clip corpus

The BD-Rate of the Pareto-Optimal curve, compared to the unmodified codec is calculated for each clip. Both SSIM and PSNR as distortion/quality metrics are used. In order to get a complete picture of the potential gains, we need to investigate if taking the *pareto* optimum of multiple encodings of a video with different values of k has a significant increase over the *direct* optimisation method detailed in the previous chapter.

The codebase for x265 was modified to take k as an argument. This allows the default λ_{orig} to be modified according to equation 2.14. We use the VideoLAN implementation of H.265, `x265`[1]. The following command invocations were used:

CRF: `x265 -input SEQ.y4m -crf <XX> -tune-<YYYY> -<YYYY> -csv-log-level 2 -csv DATA.csv -output OUT.mp4`

CBR: `x265 -input SEQ.y4m -bitrate <ZZZZ> -tune-<YYYY> -<YYYY> -csv-log-level 2 -csv DATA.csv -output OUT.mp4`

where `SEQ`, `DATA`, and `OUT` are the filenames for the raw input file, output log file and output encoded video respectively. We use the operating points as shown in Table 4.1 for CRF `<XX>` and target bitrate `<ZZZZ>`. `<YYYY>` is the distortion metric used, either PSNR or SSIM.

We use Brent's method to directly optimize each range for a given clip. Our experiments are conducted at 5 RD points as detailed in Table 4.1. Each invocation of Brent's method requires approximately twelve (12) RD-Curve generations for each bitrate range. This leads to 180 video encodes for optimisation of a given clip. Using the RD-points generated in

[1]Version: 3.0+28-gbc05b8a91

each step of the direct optimization process, we fit an exponential curve with a step size of 1kbps. We use an exponential fit to the measured RD points to then interpolate samples at the interval we require. The same exponential expression is used as for the BD-Rate calculation Bjøntegaard (2001). Using these curves we take the encode which has the maximum quality at each bitrate to form our Pareto-Optimal curve.

It is important to remember that the Pareto Optimal method uses the Direct Optimiser Results at the three proposed ranges from Table 4.1, but then extends the BD-Rate calculation by using different values for k at each bitrate using interpolated curves. In contrast the direct optimisation method uses the same k across the entire curve. This means that in practice we do not re-encode for every operating point in our range.

4.4 Results

Each graph in Figures 4.5 and 4.4 shows the cumulative distribution of the resulting maximum improvement (minimum BD-Rate using the optimal $k = k_D$). The label *Direct* refers to the system presented in the previous chapter where we directly optimize across a wide bitrate range and *Pareto* refers to the system presented in this Chapter. It is important to note that we are not comparing SSIM to PSNR nor CBR to CRF encoding, but investigating the improvement in gains by using the per bitrate system. Table 4.2 presents the key takeaways from these graphs.

Table 4.2 also reports the *average positive BD-Rate improvement* as our *Average Final BD-Rate Gain*. This is the gain measured when our system is used as a post-process or *optimized re-run* of the encoder given an initial encode with default settings. The best BD-rate between the initial encode and the re-run defines the final BD-Rate Gain. This is the most likely use of systems of this kind, and can even be considered as a two-pass system. This allows us to take advantage of any improvements in BD-Rate from Lagrangian multiplier

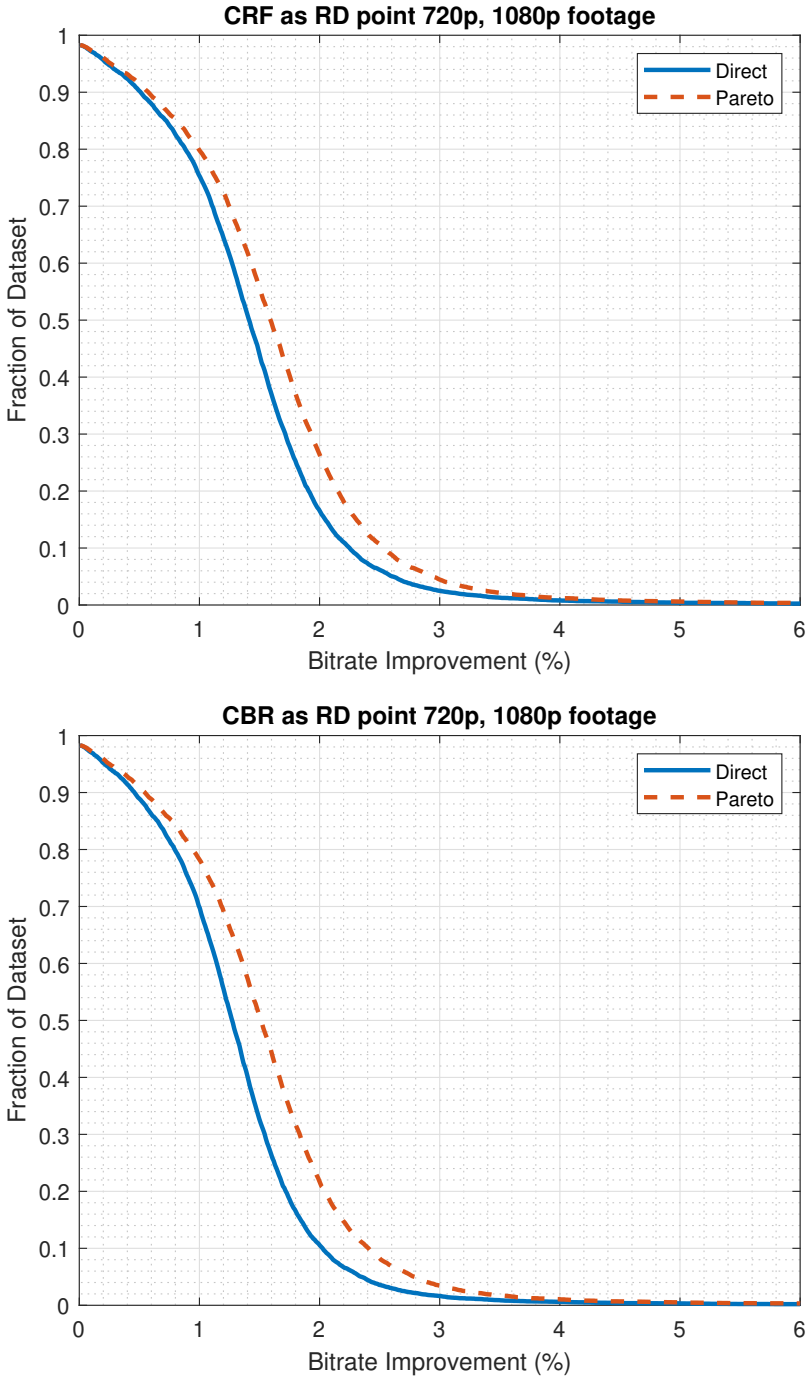

Fig. 4.4 *CDF plots of the BD-Rate improvement across our corpus using PSNR as the distortion metric. Top CRF operating point, Bottom CBR operating points. We can use these graphs to get an estimate of how much of the corpus has had BD-Rate improvements using the Direct optimizer system (average improvement 1.61% CBR, 2.24% CRF) as well as using the Pareto-Optimal Direct optimization presented in this chapter (Pareto) (average improvement 2.67% CBR, 3.03% CRF).*

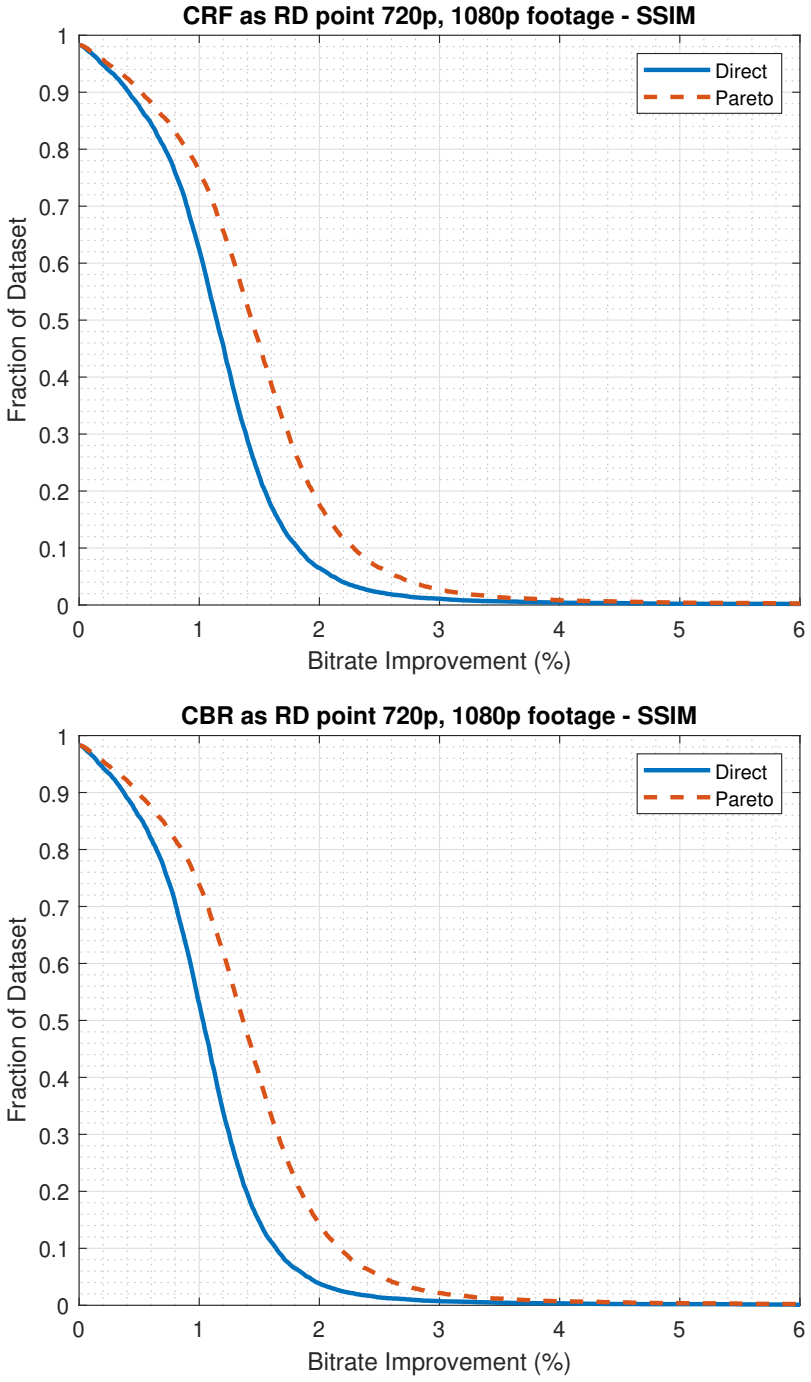

Fig. 4.5 *CDF plots of the BD-Rate improvement across our corpus using SSIM as the distortion metric. Top CRF operating point, Bottom CBR operating points. We can use these graphs to get an estimate of how much of the corpus has had BD-Rate improvements using the Direct optimizer system (average improvement 1.56% CBR, 1.05% CRF) as well as using the Pareto-Optimal Direct optimization presented in this chapter (Pareto) (average improvement 3.02% CBR, 2.24% CRF)*

Table 4.2 *Comparison of BD-Rate Gains from Direct Search and Pareto-Optimal methods. The BD-Rate is calculated w.r.t. the unmodified codec i.e.* $k = 1$. *The fourth and fifth columns show the % of clips which exhibit any improvement (*$\geq 0\%$*) and* $> 1\%$ *improvement respectively. In a practical system, the output transcode would be rejected if the compression performance is worse (BD-Rate* < 0*) than when* $k = 1$. *Hence the* ***average final BD-Rate gain*** *is reported with this rule in place. The Pareto-Optimal method has a 1-1.5% improvement when compared to the Direct Search Optimisation method. This improvement holds regardless of the quality metric and encoding mode.*

Operating Point	**Distortion Metric**	**Estimation Method**	**Clips with BD-R Gain of** $\geq 0\%$	$> 1\%$	**Avg Final BD-Rate Gain**
CBR	PSNR	Direct	96%	68%	1.61%
CBR	PSNR	Pareto	96%	79%	2.67%
CRF	PSNR	Direct	96%	74%	2.24%
CRF	PSNR	Pareto	96%	79%	3.03%
CBR	SSIM	Direct	98%	49%	1.56%
CBR	SSIM	Pareto	98%	74%	3.02%
CRF	SSIM	Direct	98%	59%	1.05%
CRF	SSIM	Pareto	98%	75%	2.34%

optimization and default to the first default encode (unmodified Lagrangian Multiplier) in the case of worse performance.

The Pareto-Optimal method from this work has approximately 1-1.5% BD-Rate improvement on average compared to the direct optimization method used in the previous chapter. This does come at the cost of at least three times the number of video encodes as the direct optimiser, as we are using three times the number of operating points to generate our interpolated curves. It appears that this method yields similar improvements, regardless of the distortion metric (PSNR/SSIM) or operating mode (CBR/CRF) used. It is possible to get further gains with more than three encoding ranges, but we believe that as we near an exhaustive solution for the optimal Lagrangian multiplier for a given clip, we would be beyond the scope of reasonable computational cost for a small bitrate improvement. This shows that there is even further potential, but at a very expensive computational cost. Because of this, it seems unreasonable to attempt to achieve further gains through pareto-optimal

optimisation, but continue using the direct optimisation of the RD-Curves from the previous chapter. The Direct Optimisation method is still quite computationally expensive, therefore the next chapter works to determine a method which reduces this cost.

Chapter 5

Reducing Computational Cost with Proxies

In the previous chapters, significant BD-Rate savings were achieved, however, this came with an increasingly large computational cost. We can use the savings found by direct optimisation as a target of what can be achieved, and furthermore, what can theoretically be achieved by finding the pareto-optimum BD-Rate improvement through Lagrangian Multiplier adjustment.

As computational complexity is of key importance given the amount of video data being processed, we explore the use of proxy systems for estimating λ at lower costs. This idea has been shown to be successful in other video processing scenarios. In particular, (Shen and Kuo, 2018) proposed a two pass rate control method in which the first pass is on a downsampled clip. They demonstrated that there is a relationship between the ideal encoding parameters, including λ, of a downsampled video and the original video. In this Chapter, we explore the use of three proxy systems. These systems deploy strategies of 1) reducing the resolution of the video, 2) changing the preset encoder settings and 3) using an older but faster video codec. This work was published in **"Per-clip adaptive Lagrangian multiplier optimisation with low-resolution proxies."** by Daniel J. Ringis, François Pitié, and Anil

Kokaram, in *Applications of Digital Image Processing XLIII*, vol. 11510, pp. 40-51. SPIE, 2020.

5.1 Proxies

Realistically, the proxy system can be anything which is expected to have similar statistical behaviour as the original clip but at a lower computational cost. These changes can be applied to the codec settings for example using I-Frames only or using the fastest preset encoders, or the changes can be applied to the input video, for example using different resolutions or frame rate. Also, we can potentially use alternate codecs which are less computationally expensive but have worse compression performance. Each of these options can potentially provide us with the appropriate adjustment to the Lagrangian Multiplier, k, while improving the computational cost. We consider three proxy systems which are detailed in the following sections.

It is important to remember that while we would be using proxy systems to determine k, the goal is to achieve an improvement in compression at the original resolution and codec. Each proxy system used is just a tool to determine k through direct optimisation at a faster rate, but the reported results would be the BD-Rate improvements when k is used in the original system.

5.1.1 Resolution Proxies (S1)

In Shen and Kuo (2018) there is evidence that codec parameters can be optimised for better compression, by learning from encoding performed at lower resolutions. To show that this is the case in our situation, we conducted on the CrowdRun sequence, from (Xiph.org, 2018) (see Figure 5.1), a direct optimisation of k at full res (k_{full}), at quarter resolution (k_{quart}) and half (k_{half}) resolution. This resulted in BD-Rate improvements of 0.33%, 0.28% and

Fig. 5.1 *Frames from the video sequence CrowdRun from Xiph.org (2018)*

0.29% respectively. Furthermore in Figure 5.2 the results show that optimal value of k is approximately the same regardless of resolution.

Taken together, these observations show that estimation of k is approximately independent of resolution. Hence we propose in our first experimental proxy system (S1), to downsample the entire corpus to 144p resolution using bi-cubic interpolation and obtain k_{144p} at that resolution. This smaller resolution represents a $\times 56.25$ reduction in the number of pixels to be processed by the codec for the 1080p clips in the corpus.

5.1.2 Faster Presets (S2)

The next proxy system we have selected is to change the codec preset options. All of our work thus far has been on the default 'medium' speed settings of x265 or the 'good' settings of VP9. We selected "ultrafast" preset for x265 and "rt" VP9 as the presets for this proxy system expecting a faster encoding time. For x265, this change in preset speed options represents a change in the CTU size, minimum coding unit size, number of B frames used, number of look ahead slices, motion vector search pattern and other encoding parameters.

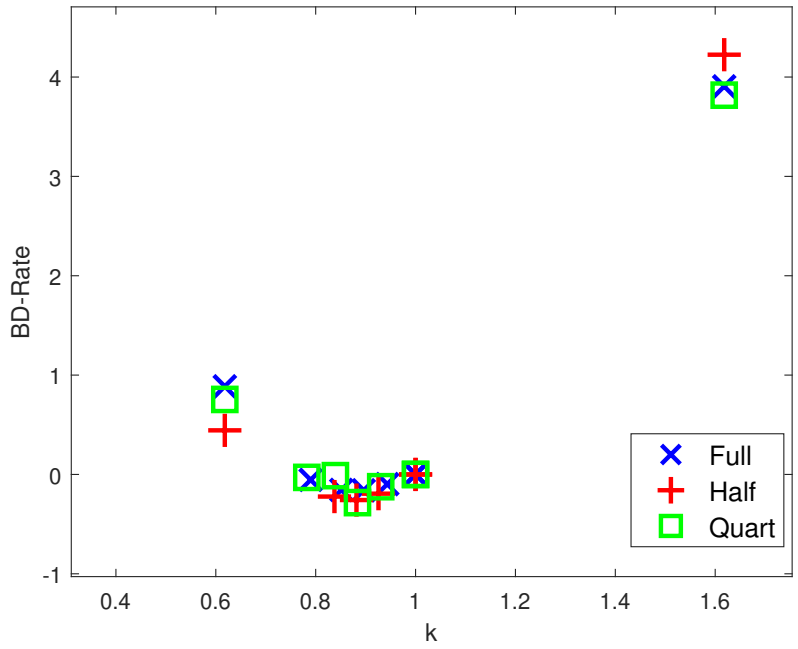

Fig. 5.2 *Direct Optimisation Search Pattern for Sequence "CrowdRun"(Xiph.org, 2018). Each marker is a value of k tested using the Direct Optimiser and the corresponding BD-Rate improvement at the resolution tested. These datapoints are the BD-Rate found for the video clip at full 'x', half '+', and quarter size '□ resolutions. We can see that the search pattern for an optimal value of k was similar across the three resolutions. In addition the minimum point is approximately the same regardless of resolution at* $k = 0.87$ *and BD-Rate improvement of approximately 0.3%. This would mean that using proxy low resolution videos can lead to similar optimal values.*

Each of these decisions would typically lead to a trade-off in compression quality for the sake of speed. We will refer to this system as S2.

5.1.3 H.264 Proxies (S3)

The third proxy system employS an older, but faster video codec to predict k similar to the work reported in Wu et al. (2021). In this case we use H.264 to estimate k in H.265 and VP9. We are aware that there are limitations in the cross-codec approach between H.264 and the newer codecs of H.265 and VP9. However, the underlying Rate-Distortion mechanism follows similar principles. We believe it is still worth investigating, as work done in Wu et al. (2021) showed that there is potential in cross codec encoding parameter selections.

We modify x264[1] in a similar fashion to x265 by inserting our own multiplier k. We refer to this cross-codec proxy system as S3 and we use the following invocations:

HEVC:

```
x265 -input SEQ.y4m -tune psnr -psnr -crf Q -frames 150
-output OUT.mp4
```

VP9:

```
vpxenc -p 1 -end-usage=cq -threads=7 -tune=psnr -psnr -cq-level=Q
OUT.mkv SEQ.y4m
```

H264:

```
x264 -frames 150 -tune psnr -psnr -crf Q -output OUT.264 SEQ.y4m
```

5.1.4 Summary of Proxy Systems

In summary, our proxy systems for the experiments are as follows:

S0: k_D direct optimisation at original resolution/codec, providing the best possible reference performance.

[1]Version: 0.157.x

Fig. 5.3 *Frames from the clip MusicVideo_360P-7b94 from Wang et al. (2019)*

S1: k_{144p} estimated at 144p and used for encoding at the original resolution

S2: k_{fast} estimated using the "ultrafast" (H265) or "rt"(VP9) setting in the codec and used for encoding at the original preset

S3: k_{h264} estimated using H264 and used for encoding in the target codec

5.2 Results

Performance of the proxy videos for the single sequence MusicVideo_360P-7b94 (Figure 5.3 from (Wang et al., 2019)) for each of the systems tested with HEVC is shown in Figures 5.4, 5.5 and 5.6. We see that in each of these figures, the proxy system provides an improvement over the unmodified codec where $k = 1$. However, the goal is to achieve a similar BD-Rate improvement to the Direct Optimiser (S0). Each RD-Curve falls short of the goal, but for this clip, the faster codec settings, S2, appear to be the closest in terms of performance as evidenced in Figure 5.5. We believe that as S2 has the least change to the codec and video being processed, it leads to the closest prediction of k.

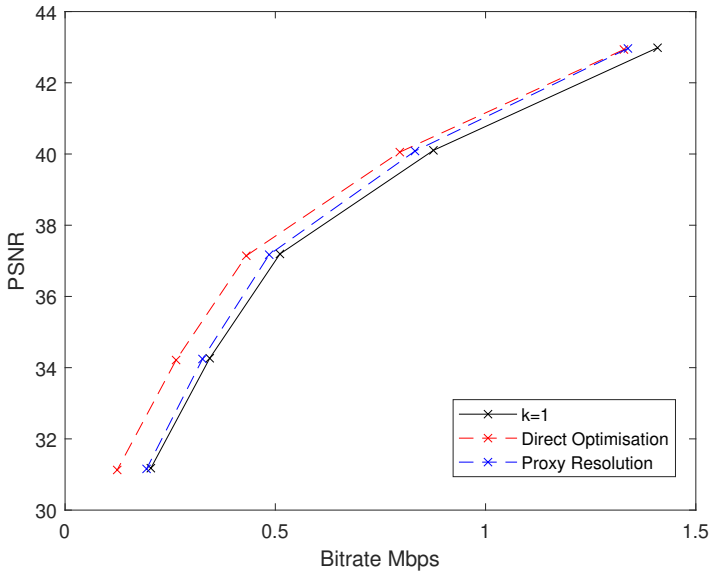

Fig. 5.4 *RD-Curves for a single sequence encoded with x265 and using our Direct Optimiser compared to the proxy video resolution encoding ie* ***S1****. We can see that for this clip, the proxy resolution provided us with a value of k which has some improvement over the unmodified codec. However, it still fell short of the target of matching the direct optimiser. Ideally, we would want the results of S1 to be equal to S0 represented by the dashed red line. This is expected to be the upper bound for performance for the proxy systems.*

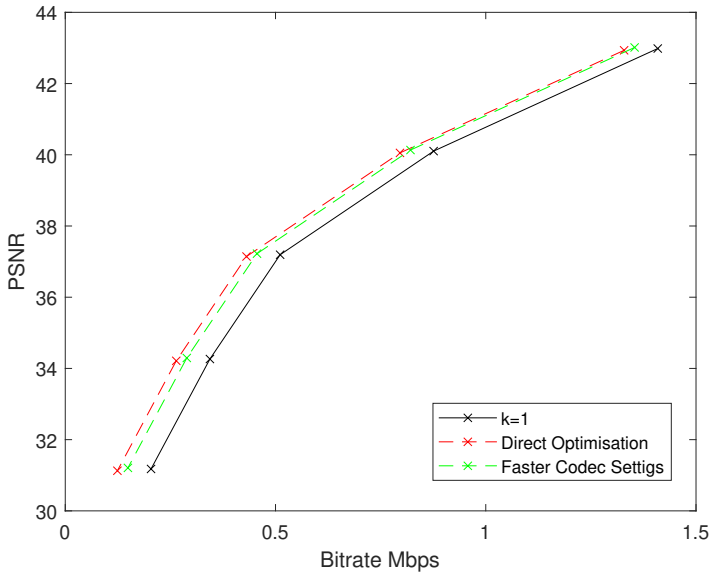

Fig. 5.5 *RD-Curves for a single sequence encoded with x265 and using our Direct Optimiser compared to using the faster codec settings as a proxy to determine the Lagrangian Multiplier Adjustment ie* ***S2****. For this clip, we can see that the performance of S2 is near equal to that of S0, which is promising.*

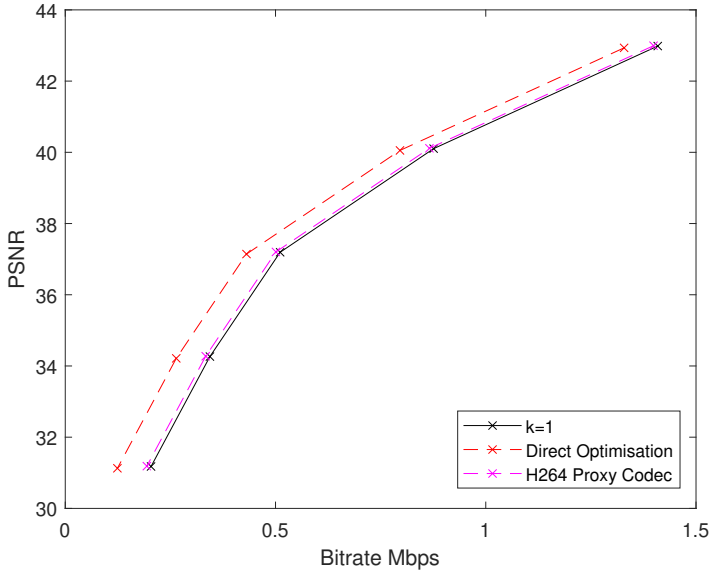

Fig. 5.6 *RD-Curves for a single sequence encoded with x265 and using our Direct Optimiser compared to using H264 as a proxy to determine the Lagrangian Multiplier Adjustment ie* ***S3.*** *Unfortunately, for this example clip, the cross codec implementation was not as successful, only providing a slight improvement over the unmodified codec with* $k = 1.0$*.*

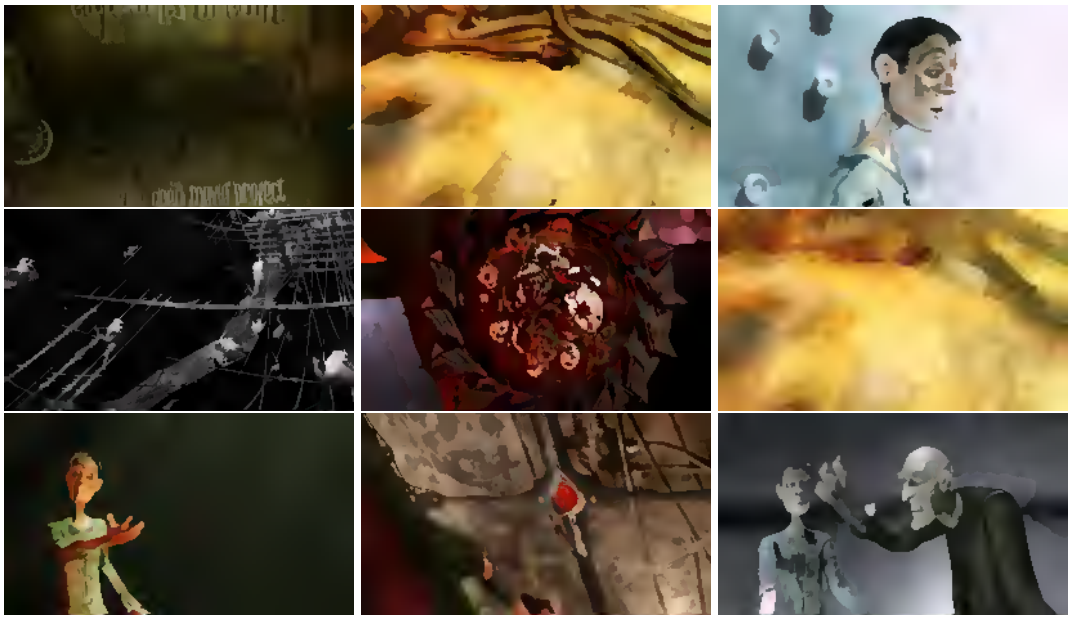

Fig. 5.7 *Sample Frames from the sequence Elephants Dream. This 10 minute movie was split into 250 DASH clips of 150 frames each for encoding using different settings. While the clips are from the same long movie, there is a lot of variety in the content of the DASH segments*

5.2.1 Speed Gains

In order to quantify the savings in compute time achieved by using a proxy we encoded 250 DASH clips (150 frames) of the sequence Elephants Dream (Figure 5.7) at 720p with each proxy. The CPU time was measured on the same machine (Intel(R) Xeon(R) CPU E3-1240 v5 @ 3.50GHz with 16GB RAM) using a single cpu thread. and the relative performance of a single iteration in each system is shown in Table 5.1. This is only an estimate for the speedup as the entire corpus is diverse in size and content.

From Table 5.1, we can see that using the smaller resolution videos gives us the biggest improvement in speed with the time to apply the direct optimiser being 22x faster with x265 and 9x faster with VP9. Encoding using x264 also shows a significant improvement in encode time with 3.7x and 9.3x (HEVC, VP9) times improvement. The smallest savings are found when using the faster encoding presets for each codec with 3.1x and 1.1x (HEVC, VP9) improvements being found using the “ultrafast” and “rt” presets respectively. If we are

Table 5.1 *Processing time comparison of VP9 and H265 with faster codec settings and smaller resolutions as well as H264 for 250 DASH clips selected from*

System	Time per clip (s)	Speedup
HEVC: S0	19.48	1
HEVC: S1	0.89	21.94
HEVC: S2	6.34	3.07
HEVC: S3	5.39	3.61
VP9: S0	50.29	1
VP9: S1	5.62	8.94
VP9: S2	47.18	1.07
VP9: S3	5.39	9.33

able to get comparable BD-Rate savings using these proxies it would indicate a significant improvement in compute time for the direct optimisation methods applied.

These speed ups are reported for just for a single movie, but they should be representative of our corpus. It is expected that the compute times would change based on the platform used. Nevertheless, we see that the use of proxies has a significant positive impact on computational load.

5.2.2 BD-Rate Improvements

Figure 5.8 shows the relationship between k estimated from direct optimisation, k_D and that estimated for various proxies k_{ALT}. For the majority of clips there is a close correlation between ground truth $k = k_D$ generated at original resolution and k_{ALT} generated at 144p. This is because the scatter plots are close to the line $k_{ALT} = k_D$. This holds true for all three systems S1, S2 and S3 for both HEVC and VP9. k is most similar to the original estimates when using the faster codec settings at the original resolution (S2) and least similar using H264 (S3). This bodes well as it shows that it is possible to provide the good performance at low cost.

Figure 5.9 summarises the BD-Rate performance gains over the whole corpus. The Cumulative density distribution of each system w.r.t. BD-Rate is reported there. Better

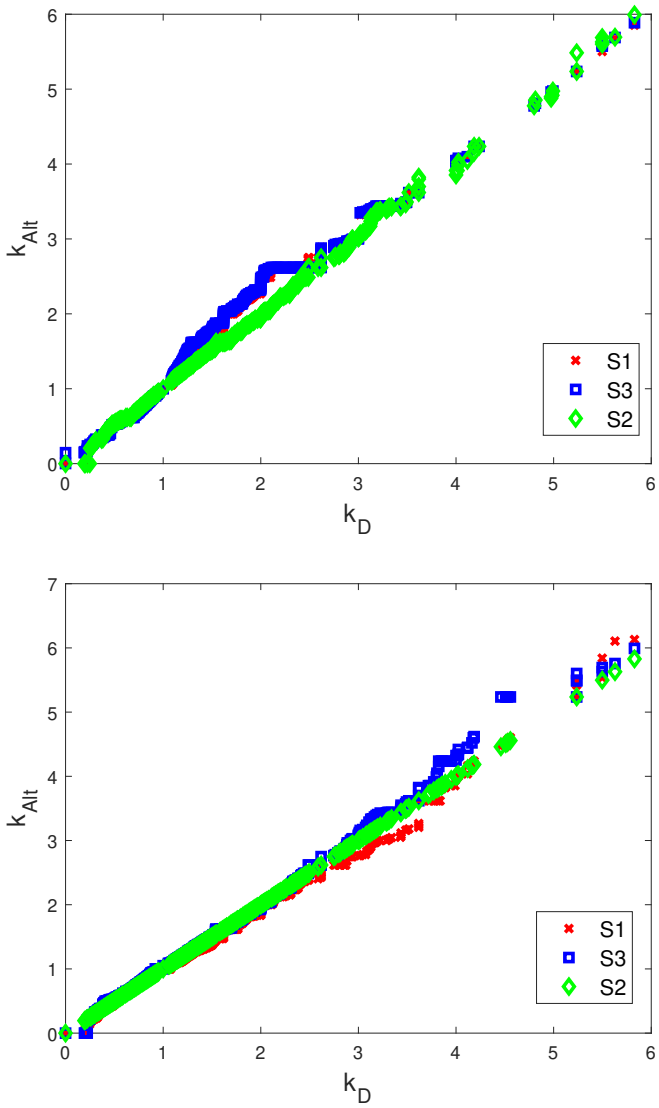

Fig. 5.8 *k_D vs $k_{\texttt{Alt}}$ determined from a proxy video or codec for x265 (top) and VP9 (bottom). Ideally, we would want points to lie on the line $k_{\texttt{D}}$=$k_{\texttt{Alt}}$ We can see a strong correlation between k and $k_{\texttt{Alt}}$ estimated by **S2** (green) and **S0**.*

performing systems are towards the top right of the plot. Using the fast presets at the original resolution (S2) gives us very similar performance to our original direct optimisation algorithm (S0) with average BD-Rate improvement of 1.54% for S2 and 1.87% for S0 in HEVC and 1.32% for S2 and 1.63% for S0 in VP9. Using a different codec (S3) to estimate k however, gave the worst results with only 0.58% (HEVC) and 0.5% (VP9) average BD-Rate Gain.

Using faster video encodes but at the original resolution (system **S2**) yields almost identical performance to ground truth but with improvement in speed of 6% for VP9. The other curves show use of smaller resolutions and/or H264 as the codec. They are much faster (up to $\times 22$) but as can be seen achieve about only 33% of the possible gains on average. However, using HEVC, the faster codec settings provide us with the best gains at a reasonable speedup. The speedup using the proxy resolution (S1) is the best, however it resulted in a much lower average gain (0.54%) across the corpus compared to 1.87% with S0.

A summary of our findings using systems S1, S2, S3 can be found in Table 5.2. It is positive to note that majority of the clips which had some improvement using the Direct Optimisation system (S0) continue to show improvement using proxy videos. To illustrate this, note that using per clip Direct Optimisation (S0), only 8% of the corpus yields no improvement in HEVC and we see a similar result in each proxy system where only 9% of the corpus yields no improvement for each proxy system. This is also seen in VP9, for all systems tested, S0-S3, 22% of the corpus have no improvement.

The ideal scenario is that the proxy systems provided us with identical results to the full cost direct optimisation system. However, the systems proposed in this chapter are effectively a trade-off between computational cost and BD-Rate improvement. We can see this represented in Figure 5.10. In Table 5.2, we list the percentage of clips which have at least 0.1% and 1% BD-Rate gain as well as the best and average gains. As the shapes of the curves in Figure 5.9 indicate, the percentage of our corpus with $> 1\%$ gain are near equal for S0 and S2. However, S2 is the slowest of the three proxy systems tested. If the goal is to maximise compression performance, it appears that continuing to use the same codec and

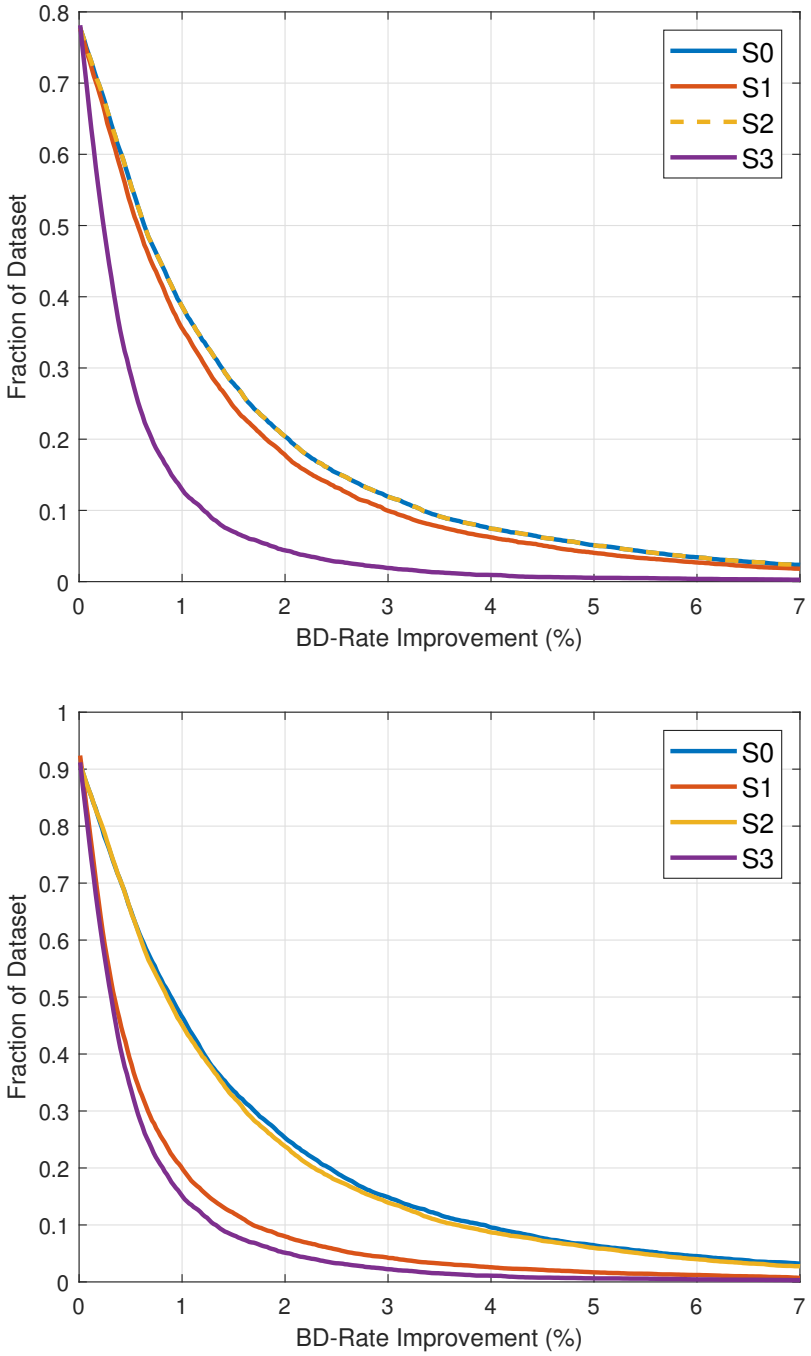

Fig. 5.9 *Cumulative distribution of BD-Rate improvement over 9,746 clips using k calculated from different proxy systems S1-S3 VP9 (top) and HEVC (bottom). The best performing systems would be nearest to the top right of the plot. In blue (S0) is the BD-Rate distribution using the original resolution and k_D i.e. the ground truth best performance possible. Using faster video encodes but at the original resolution (system* **S2** *yellow) yields almost identical performance to ground truth but with improvement in speed of 6% for VP9 and 300% for HEVC. The other curves show use of smaller resolutions and/or H264 as the codec. They are much faster (up to* $\times 22$*) but as can be seen achieve about 33% of the possible gains on average.*

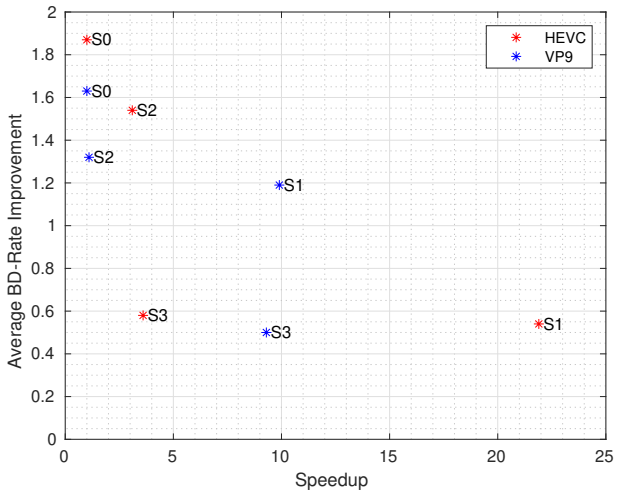

Fig. 5.10 *Speedup vs Average BD-Rate Gain for the proxy systems. Ideally, we would like each proxy to achieve BD-Rate improvements comparable to S0 for each codec. We can see an inherent trade-off between the gains found and the speedup for the proxy systems.*

video resolution is best. On the other hand, if the goal is to get any BD-Rate improvement as fast as reasonably possible, using the reduced resolution videos appears to be ideal. With speedups of approximately 22× and 9× for HEVC and VP9 respectively, a significant portion of the corpus achieves at least 0.1% BD-Rate improvement. Overall, choosing which of these systems provides the best performance will be based on the preference for speed vs compression. From Figure 5.10 it seems that system S1 provides an acceptable trade-off for VP9, while a clear winner does not appear for HEVC.

For system S3, it is unsurprising that the results obtained from H264 do not perform well when inserted into VP9, there is less overlap between the two codecs than say H264 and H265. This is similar to the experience reported by Wu et al. (2021), who indicated they had better results using VP9 with AV1 in their cross codec scheme. However, it is very positive to see how close the reduced resolutions and faster codec settings are to our original system while being less computationally expensive.

Table 5.2 *Summary of* **BD-R Gains** *results using Direct Optimisation with Proxies. This shows the % of clips which have no improvement,* 0.1% *and* 1% *improvement, as well as the best and average BD Rate improvement.*

System	Clips with BD-R gain of			Best	Avg	Speedup
	$= 0\%$	$> 0.1\%$	$> 1\%$	Gain	Gain	
S0: x265	8%	87.00%	46.24%	23.86%	1.87%	$\times$1.0
S1: x265	9%	78.17%	18.75%	20.46%	0.54%	$\times$21.9
S2: x265	9%	86.57%	45.23%	20.59%	1.54%	$\times$3.1
S3: x265	9%	74.21%	14.13%	20.12%	0.58%	$\times$3.6
S0: VP9	22%	74.00%	38.75%	22.13%	1.63%	$\times$1
S1: VP9	22%	73.00%	35.73%	20.60%	1.19%	$\times$8.9
S2: VP9	22%	74.00%	38.75%	22.13%	1.32%	$\times$1.1
S3: VP9	22%	66.00%	12.85%	19.36%	0.5%	$\times$9.3

5.3 Conclusion

In the proposed systems, the majority of clips show some BD-Rate improvement. The average BD-Rate improvement is in the range of 1% across the corpus. We see in Table 5.2 that the best tradeoff is system S1 for VP9 (1.19% BD-Rate avg gain out of a possible 1.63% with S0, thus 73% of the potential average BD-Rate gains at $\times$ 8.9 speed) and S2 for HEVC (83% of the possible average BD-Rate gains from S0 at $\times$ 3.1 speed).

Another option to be considered is combining these different proxy methods to achieve even further speedup. The idea is that combining the use of the smaller resolution video with faster codec settings or even doing this using the proxy video codec can achieve a further increase in speed. However, as each individual system only achieved a fraction of the compression performance of the direct optimiser, it was decided that only focusing on one form of proxy video to determine the appropriate Lagrangian Multiplier adjustment was best.

Using the direct optimisation methods still requires a large number of video encodes for a given clip regardless of resolution. Therefore there is still a need to reduce computational cost even with proxies. In the next chapter we explore the idea of using ML to estimate k using low complexity features extracted from the video clip.

Chapter 6

Reducing Computational Cost by Prediction

One way to improve speed is to reduce the number of pixels being processed, or use alternate codecs which are faster. These kinds of “proxies” were explored in Chapter 5. The other approach is to predict k from a set of easily computable or already available features, for example the target bitrate, measured input bitrate or I and P frame properties. This sort of idea was proposed by Covell et al. (2016) for another purpose. In this Chapter, we explore prediction through using various feature sets with different sorts of machine learning approaches. This work was published in:

- **"Per-clip adaptive Lagrangian multiplier optimisation with low-resolution proxies."** by Daniel J. Ringis, François Pitié, and Anil Kokaram, in *Applications of Digital Image Processing XLIII*, vol. 11510, pp. 40-51. SPIE, 2020
- **"Near optimal per-clip lagrangian multiplier prediction in HEVC."** by Daniel J. Ringis, François Pitié, and Anil Kokaram, in *2021 Picture Coding Symposium (PCS)*, pp. 1-5. IEEE, 2021 and received a **Top 10 paper award.**

6.1 Prediction Methods

The direct optimisation method outlined in the previous chapters, provides positive results, however, it is computationally expensive. It takes on average 11.7 iterations, and, as each BD-rate computation takes 5 video encodes, finding the optimal k takes $11.7 \times 5 \approx 60$ video encodes. This is manageable at small resolutions, but is computationally intensive at modern resolutions (720p and higher) and when processing thousands of videos or more. Because of this, we have proposed in Chapter 5 to reduce the computational cost by using proxies (low resolutions, older codecs) to calculate k in practice. We were able to get a reduction of $3.1\times$, while achieving 83% of the gain found by direct optimisation.

In order to further reduce the computational cost, it is sensible to consider machine learning and deep learning systems which make use of features of the video to predict k.

6.1.1 Features

For prediction, it is important that we use features that require lower compute overhead to extract. We examine two overarching classes of low complexity features. These are features provided by the encoder, and features related to the content of the video.

Encoding Features Initially, we used a limited feature set of features available to us from the encoder. This was done as a basic "is this possible?" check, before spending time trying to tune, optimise and ensure that the best feature set was selected.

Our first proposed feature set consists of 17 features based on the available information from the encoding logs: Overall Bitrate, Overall PSNR, PSNR for Y, U and V channels, Avg bitrate for I, P and B frames, Avg PSNR (Y,U,V) for I, P and B frames, Frame height and width, YouTube video class. The latter classification is encoded as a numeric value mapped from the class provided by YouTube.

Similar to Covell et al. (2016), non linear extensions of the feature set were created from products of feature elements by taking the product of a subset of feature pairs. This feature

Table 6.1 *Expanded list of Encoding Features used in our system. These features are used in addition to the feature vector listed in Appendix 2*

Features Per Clip	Features Per Frame
Average Bitrate	Bits
Average PSNR (Y, U, V, Global)	I/P cost Ratio
Average I, P and B frame count	PSNR (Y,U,V)
Bitrate for I, P and B frames	Avg Chroma Distortion
I Frame PSNR (Y,U,V)	Average Residual Energy
P Frame PSNR (Y,U,V)	Avg Luma Level
B Frame PSNR (Y,U,V)	Avg Cb Level
	Avg Cr Level

vector therefore contained 49 elements. The full list of these features is available in Appendix 2. We refer to this feature set as our "encoding features".

Expanded Encoding Features The original features tested were limited and initial experiments (detailed in a later section) showed promise, but were not as successful as we had hoped. Because of this we expand our feature set by including the per frame features. See Table 6.1 for the additional features, the full list is available in Appendix 2. This greatly expands the feature vector from 49 to 1523 features. The features are collected for a clip encoded with no modifications to the codec, at CRF 32. While it is possible that not all of these features are required, they are collected at no additional computational cost, as they are all generated from the same video encode from the unmodified codec. This is referred to as our "expanded encoding features".

DNN Features We observed, indeed, that the nature of the clips content influenced the Lagrangian optimisation. Hence, our last set of features uses the VGG16 network (Russakovsky et al., 2015) as shown in Figure 6.1 to generate semantic visual features from a single frame in each video sequence. The 1000 features from the fc7 layer of VGG16 are used on a downsampled (224×224) midpoint (75th) frame of the sequence. This is referred to as our "VGG Features".

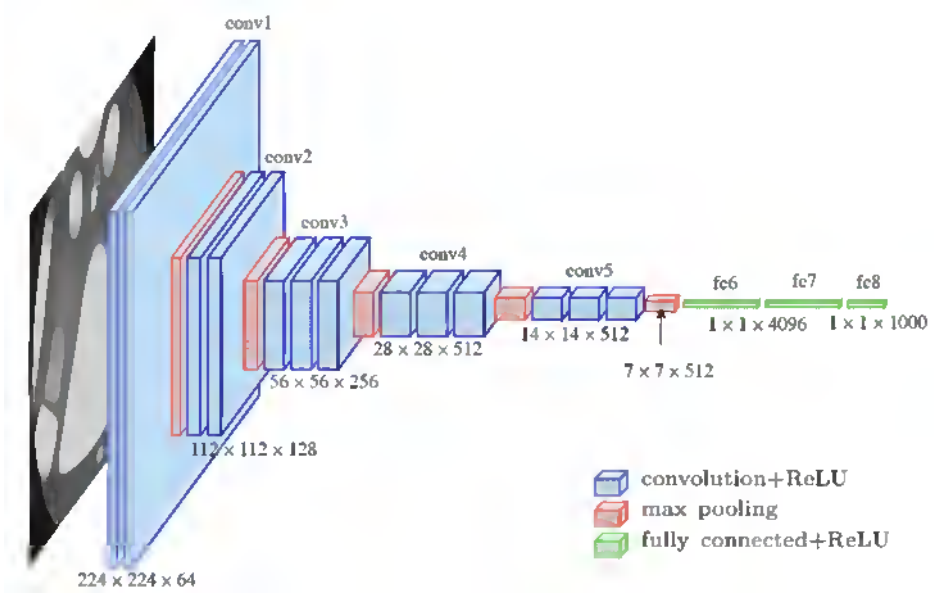

Fig. 6.1 *The VGG16 Network, taken from Russakovsky et al. (2015), used to provide additional features about the video being encoded. We take the results from the fc7 layer and feed that feature vector into our network*

6.1.2 Random Forest Model

We investigate an offline process in which the first step measures video features that are mapped to k through an ML algorithm. This significantly reduces computational complexity. In essence we are generalising the previous approaches to reach beyond regression in estimation of k.

We use k_D as ground truth for training our ML process. Note that k was range limited $0 < k < 3$ and we hold out 10% of the dataset for testing. With the remaining corpus, cross validation with five (5) folds was done for the training to minimize over-fitting. We tested 19 ML methods (including Logistic and Linear Regression SVMs, Decision Trees) using MATLAB2020 Machine Learning Toolbox and found that Random Forests yielded best results. The Random Forest algorithm with 8 leaves, 30 learners was the best performing ML system tested. The best outcome for these experiments would be to find that prediction through Machine Learning provides the same performance w.r.t. BD-Rate as Direct Optimisation. Details about the performance of this algorithm are listed later in this chapter.

Table 6.2 Details of the Dense Layer sizes (n), dropout (0.1) and batch-normalisation (BN). All layers use the Gaussian Error Linear Unit (GELU) activation function.

layer id	1	2	3	4	5	6	7	8	9	10	11
n	1000	800	600	200	100	64	32	16	8	4	1
BN	✓	✓									✓
Dropout	✓	✓	✓					✓			

6.1.3 Deep Learning Model

Two feature vectors (Expanded Encoding Features and VGG Features) are concatenated into a combined feature vector of size 2523 for our model. Using the VGG features and NN architecture does come at a cost of increased complexity, but this may be preferable to multiple video encodes required for the direct optimisation method.

Our model consists in a sequence of fully connected layers with GELU activation functions and Normalisation Layers. A full breakdown of the layers can be found in Table 6.2. The model was trained for 5000 epochs, using the SGD optimiser, a learning rate of 0.001 and a momentum of 0.9. The loss function is the mean absolute error between the predicted and directly optimised k. We can see this graphically represented in Figure 6.2.

6.1.4 Training

We again use the entire corpus mentioned in Chapter 3. The corpus is split 70%/30% between training and testing. In order to ensure that the test and training set did not have any overlap, different DASH segments from the same sequence were not placed in both the testing and training sets.

6.2 Experiments

Our first experiment focuses on using Random Forest implementation using the limited set of encoding features . We use our dataset of approximately 10,000 clips from a variety of

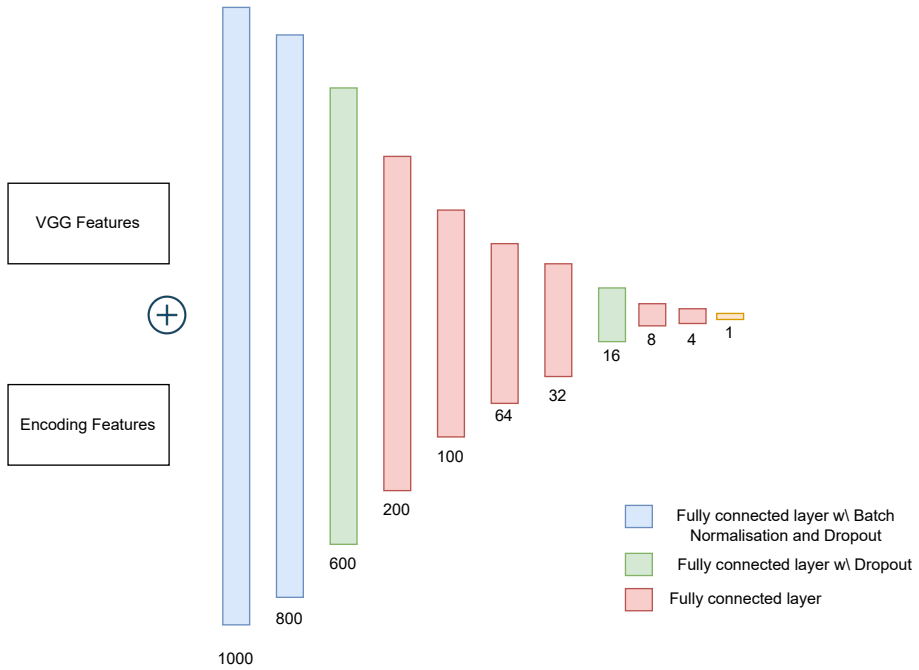

Fig. 6.2 *Diagram of the layers for the fully connected network used to determine k from the encoding and VGG features*

sources as used in the previous chapters (detailed in Chapter 3 and Appendix 2). We test three systems, once using the original resolution and codec, the others use the proxy systems implemented in the previous chapter. It was decided that using k from a different codec, such as H264, did not have a significant enough impact in the previous section to continue with that experiment in a machine learning system. This leads to three ML systems to be examined as follows.

ML0: k_{M} using features estimated using the Random Forest Model at original resolution/codec

ML1: k_{M144} using features estimated at 144p using the Random Forest Model and used for encoding at the original resolution

ML2: k_{Mfast} using features estimated using the "ultrafast" (H265) or "rt"(VP9) setting in the codec using the Random Forest Model and used for encoding at the original resolution

These are compared to the direct optimisation scheme (S0) detailed in Chapter 4. S0 is expected to represent the upper-bound of improvements we can achieve per clip.

6.2.1 Using the Expanded Feature-set

While there is promise in using the proxy videos in the machine learning systems, the original size videos used in ML0 show the best results. Because of this we limit the work using the expanded feature-sets (both the expanded encoding features and the use of VGG features) to predictions without the use of proxies.

While there is significant improvement to be had in terms of speed by using the reduced resolution video, it is expected that the encoding features at the original resolution and codec settings would be best suited to developing a machine learning model for this work.

6.3 Results

Using ML0, the random forest model with the encoding features, we achieve 64.74%, 62.38 % (HEVC, VP9) of the possible BD-Rate gain at original resolution (i.e. S0). Furthermore, 66.96%, 58.22 % (HEVC, VP9) of the video corpus showed an improved BD-Rate. Similar results were found with ML2, which used the results from the ”ultrafast“ encodes with 63.48%, 57.79 % (HEVC, VP9) of the possible BD-Rate gain at S0 were achieved.

The biggest savings in computational time would be ML1, where we extract features at 144p, and use these to estimate k, (k_{Msmall}), and encode at the original resolution. We find that 56.82%, 54.27% (HEVC, VP9) of the video corpus showed a BD-Rate improvement. Overall this implied that 64.24%, 67.72% (HEVC, VP9) of the possible gain was achieved.

Table 6.3 shows a summary of the findings for systems ML0, ML1 and ML2. With nearly half of clips showing no improvement, we can see that this needs more work. Determining what features of the video are best suited to predict the appropriate k for RD optimisation is a challenge. However, this work shows promise by being approximately six (6) times faster

Table 6.3 *Summary of* **BD-R Gains** *Results using Machine Learning with Proxies. This shows the % of clips which have no improvement(including those which performed worse), 0.1% and 1% improvement, as well as the best and average BD Rate improvement*

System	Clips with BD-R gain of			Best	Avg	Speedup
	$= 0\%$	$> 0.1\%$	$> 1\%$	Gain	Gain	
ML0: x265	33%	27.5%	2.2%	8.54%	0.19%	$\times 6$
ML1: x265	43%	26.8%	2.1%	11.83%	0.18%	$\times 131$
ML2: x265	37%	26.6%	2.4%	13.93%	0.19%	$\times 19$
ML0: VP9	41%	44.1%	3.1%	9.62%	0.14%	$\times 6$
ML1: VP9	45%	39.7%	3.2%	10.09%	0.14%	$\times 53.4$
ML2: VP9	42%	40.1%	3.3%	8.41%	0.14%	$\times 6.6$

than the direct optimisation methods in Chapters 3 and 4 as only two RD-Curves are required to predict k for a given clip.

6.3.1 Expanded Feature-Set

We repeat this experiment using the ML0 model i.e. without the use of proxies. In addition we employ the expanded encoding features with the VGG features as presented in Section 6.1.1. This feature-set is used for both the random forest model and our proposed deep learning model. We see in Figure 6.3, how each model fares compared to Direct Optimisation (S0) as well as the best global improvement $k = 0.782$ w.r.t. BD-Rate improvement for HEVC.

In these graphs, as well as the results presented in Table 6.4, we see that roughly 95% of the clips had some BD-Rate improvement, with 46% of the corpus having a 1% or better BD-Rate improvement from direct optimisation. Our machine learning systems (Deep Learning and Random Forest) give positive results with 89% and 69% of the corpus getting a BD-Rate improvement but only 39% and 28% of the corpus getting a 1% or better BD-Rate improvement. We also see in Figure 6.3 that both machine learning systems drastically outperform the global Lagrangian Multiplier Adjustment which once again, confirms the benefits of using a per clip approach.

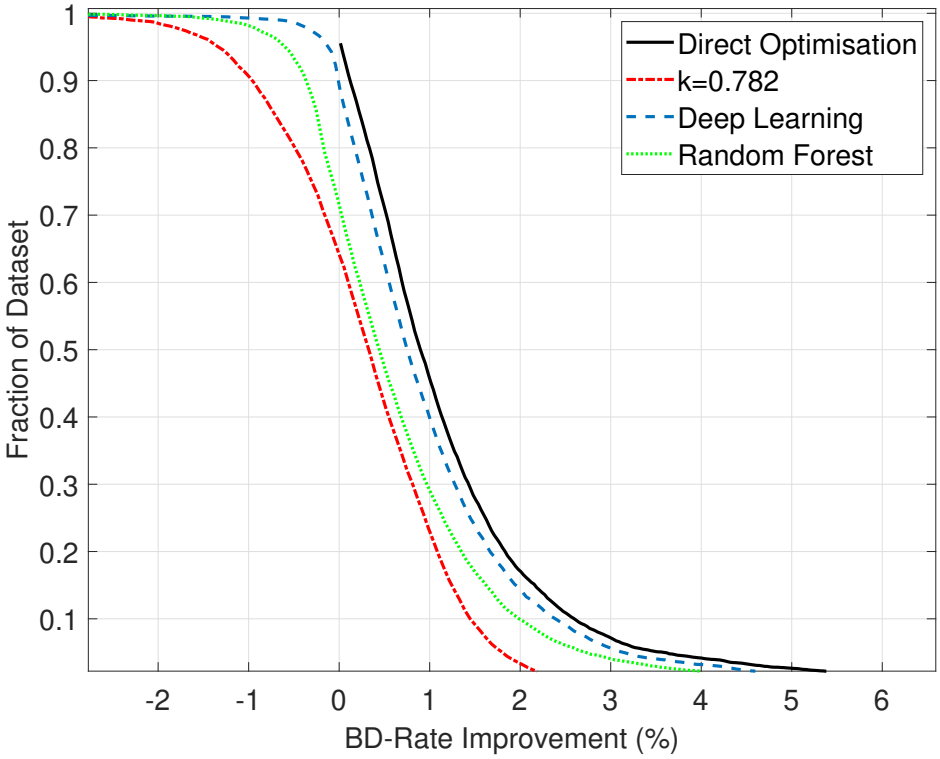

Fig. 6.3 *CDF plots of the BD-Rate improvement across our corpus for Random Forest and our proposed DNN method using the expanded encoding features and VGG features. We see that both per-clip approaches outperform an overall adjustment of the Lagrangian multiplier ($k = 0.782$) and that our network reaches near optimal performance.*

Table 6.4 *Summary of* **BD-R Gains** *results for HEVC. This shows the % of clips which have any improvement,* 0.1% *and* 1% *improvement, as well as the best and average positive BD Rate improvement*

Method	**Clips with BD-R Gain of** $\geq 0\%$	$> 0.1\%$	$> 1\%$	**Best Gain**	**Avg Final Gain**
Direct Optimisation - S0	95%	89%	46%	23.9%	1.87%
Random Forest - ML0 (encoding features)	67%	28%	2%	8.5%	0.19%
Random Forest -ML0+ (expanded encoding & VGG features)	69%	65%	28%	18.3%	0.82%
$k = 0.782$	62%	56%	23%	23.5%	0.63%
DNN (encoding & VGG features)	89%	84%	39%	14.9%	1.11%

Table 6.4 also reports the *average positive BD-Rate improvement* as our *Average Final BD-Rate Gain*. This is the gain measured when our system is used as a post-process or *optimised re-run* of the encoder given an initial encode with default settings or more simply put, the worst BD-Rate possible with this measurement is BD-Rate $= 0\%$ when $k = 1$. The best BD-rate between the initial encode and the re-run defines the final BD-Rate Gain. This is the most likely use of systems of this kind, and can even be considered as a two-pass system. This allows us to take advantage of any improvements in BD-Rate from Lagrangian multiplier optimisation and default to the first default encode (unmodified Lagrangian Multiplier) in the case of worse performance. In this situation, we can see that our deep learning approach almost achieves what the direct optimiser does, at a fraction of the computational cost. We see that 89% of the corpus has some improvement using the DNN compared to 95% through the Direct Optimiser (S0) and similar fractions of the corpus show at least 0.1% (89% for S0, 84% for DNN) and 1% (46% S0, 39% DNN) BD-Rate improvement for the two systems. The direct optimiser would have required on average sixty video encodes (approximately

12 RD-Curves with five operating points each) whereas our Deep Learning approach only required ten (2 RD-Curves, baseline and estimated k) to get near comparable improvements.

6.4 Conclusion

In this chapter, we have presented further data to support a per clip approach for the Lagrangian Multiplier within the HEVC codec. We build upon our direct optimisation results, by implementing a deep learning model capable of comparable results at a fraction $(1/6)$ of the computational cost. 90% of our corpus shows some BD-Rate improvement with deep learning (compared to 95% of the corpus with direct optimisation). We also see that we can get approximately 1% BD-Rate savings across our entire corpus if we use the deep learning system as a supplement to the existing codec. In summary, our deep learning approach nears the direct optimiser results while using only $1/6^{th}$ of the computational cost. It is possible that training our models using $BD-Rate$ as the loss function would lead to better results as opposed to using the internal parameter k. Therefore, future work will include investigating the deployment of this system for quality/distortion improvements at certain bitrates as well as BD-Rate gain directly as well as expansion to newer codecs like VVC.

Chapter 7

Conclusion

This thesis presents a novel approach to optimally parameterising established modern video codecs through the use of classical numerical techniques such as direct optimisation. With the continuous growth of video streaming platforms such as YouTube, Netflix, Disney+, video content comprises up to 80% of internet traffic. There is an increasing demand for high quality video at low bitrate. This chapter summarises the impact from the approaches we made to improve the compression in modern video. For this thesis we proposed two main research questions:

Research Question 1: What are the potential gains achievable by per clip optimisation of the Lagrangian Multiplier compared to the current implementation in common video codecs ?

Research Question 2: Can we achieve the potential gains at a reduced computational load through the use of proxy systems or prediction?

7.1 Direct Optimisation of Rate Distortion Optimisation

In order to answer the first research question, we established that a per clip approach is best suited for improving the Rate Distortion Optimisation modules of video codecs. For

our corpus of approximately 10,000 video clips, we were able to get an average BD-Rate improvement of 0.63% for HEVC using an updated value for the Lagrangian Multiplier, λ, where $\lambda = 0.782 \times \lambda_{orig}$. This global re-engineering of the Rate Distortion Optimisation within the codec highlights that there is inefficiency in the current implementation. However, roughly a third of the clips do not respond well to this change, and have worse compression performance than the unmodified codec. This indicated that the global approach to adjusting λ within the RDO module within the codec is not optimal.

We therefore adopted a per clip direct optimisation approach to the RDO module within HEVC and VP9. Using a single parameter k allowed for the re-engineering of the Lagrangian Multiplier where our new Lagrangian Multiplier, $\lambda_{new} = k \times \lambda_{orig}$. By doing this, the average BD-Rate improvement for our corpus was approximately 2% through Direct Optimisation on HEVC. Our best BD-Rate improvement through Direct Optimisation was 23.86% for HEVC and 27.05% for VP9. There can be a great impact in the field of video streaming, if uploaded videos are streamed in the thousands to millions of views range, a bitrate savings of as small as 0.1% can have an impact. Over 70% of the corpus had a BD-Rate improvement of at least 0.1% for either codec tested.

Also, optimising video codecs using a corpus which best represents today's videos, leads to BD-Rate improvements. By using the video corpus of almost 10,000 video clips, majority of which are user generated video content, we can find potential positive outliers in improvement. The videos which have low temporal complexity, which are fairly commonly uploaded, such as animation clips, lecture slideshows and lyric videos seem to have significant improvements when the RDO modules within the codec are adjusted.

We also investigated a Pareto-Optimal approach to determining k for a single clip by estimating the best improvement per bitrate after directly optimising w.r.t. BD-Rate for three ranges (Low, Medium and High). This Pareto-Optimal method from this work has approximately 1-1.5% BD-Rate improvement on average in HEVC compared to the direct optimization method used in the previous chapter. This does come at the cost of at least three

times the number of video encodes as the direct optimiser, as we are using three times the number of operating points to generate our interpolated curves.

The direct optimisation system however, comes at a high computational cost. even with a fairly relaxed stopping criterion and using five operating points per rate distortion curve, each video would need to be encoded approximately sixty times in order to find the improvement through direct optimisation. We then address this high computational cost with two overall methods applied to reduce this cost.

7.2 Computational Cost Reduction

The two approaches taken to reduce the computational cost of the direct optimiser were reduction through the use of proxies and reduction through prediction. We use the improvements found by the originally proposed direct optimisation system as the target level of improvement. This effectively targets our second research question.

We have presented three proxy systems which apply the direct optimisation algorithms but at a reduced cost. The three methods of reducing the computational cost to compute k, were using 144p versions of the input video, using the ultrafast presets of the codecs and using H.264, as an older but faster codec. Of these, the best improvement in speed came from using the reduced resolution video clips and the compression performance closest to the full scale direct optimiser was achieved by the faster presets.

There exists a trade-off between speed and compression performance, as the speed of the system increases, the compression performance tends to decrease which we can see in Figure 7.1. The best trade-off is the reduced resolution for VP9 (1.19% BD-Rate avg gain out of a possible 1.63% with direct optimisation, thus 73% of the potential average BD-Rate gains at $\times$ 8.9 speed) and faster presets for HEVC (83% of the possible average BD-Rate gains from direct optimisation at $\times$ 3.1 speed). It is important to note that this work does not aim to compare HEVC and VP9, but to improve both codecs using the same methods, with

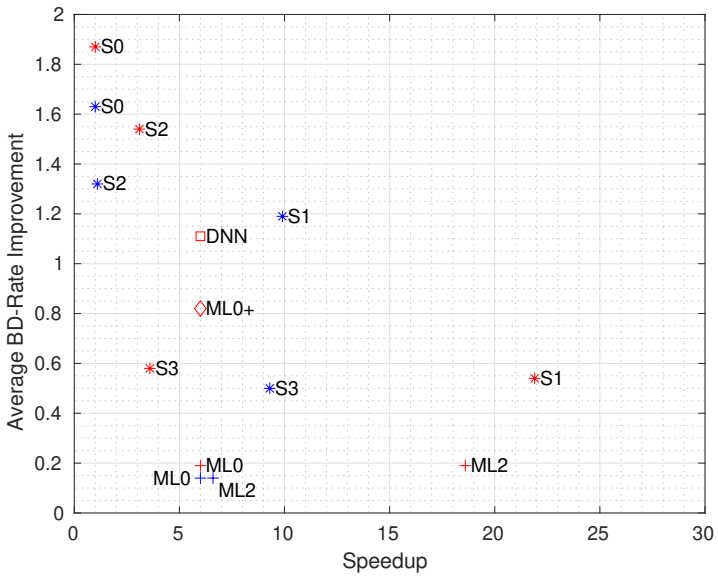

Fig. 7.1 *Speedup achieved vs Average BD-Rate Gain for each system tested in this thesis. HEVC systems are in red and VP9 systems are in Blue. Note that systems ML1 for both codecs are beyond the limits of this graph:* $> \times 40$ *speedup improvement at BD-Rate* < 0.2*. We can see that the Direct Optimisation system (S0) achieves the best BD-Rate improvement. We note that for VP9 using smaller resolutions leads to an acceptable BD-Rate achieved at a considerable speedup. Similarly, we can see that our DNN system for HEVC provides an acceptable trade-off in BD-Rate performance and speedup compared to the Direct Optimiser*

respect to their default configurations. We find that using proxies is useful and in fact can achieve comparable results at much faster speeds.

This reduction in computational cost is significant. It still is dependent on multiple video encodes but using proxy videos to determine what adjustment to be made leads to at least comparable gains with a more efficient method. This showed that there is promise in looking at the content or features of the video, and the encoding of the video to determine the adjustment to the Lagrangian multiplier in Rate Distortion Optimisation in video codecs.

The other method which reduced the computational cost was through machine learning. Namely, a Random Forest Algorithm and a Deep Neural Network. We used features of the video collected on a first pass encode of each clip, such as PSNR and bitrate for the different colour channels channels and different frame types. Initial experiments with just the encoding information showed promise, with 20-30% of our target gains being achieved with

our random forest implementation to determine k for per clip rate distortion optimisation. However, the closest performing system to our direct optimisation system was when we introduced image features from the well established VGG model. We achieved almost 70% of our target bitrate savings with deep learning. This was done with just two Rate Distortion Curves worth of operating points being encoded. We can see a summary of the speedup achieved w.r.t. the time and number of video encodes required to estimate k vs average BD-Rate gain in Figure 7.1. We note that for VP9 using smaller resolutions (S1) leads to an acceptable BD-Rate achieved at a considerable speedup. Similarly, we can see that our DNN system for HEVC provides an acceptable trade-off in BD-Rate performance and speedup compared to the Direct Optimiser.

7.3 Corpus Consideration

Much of the previous work in this space used small corpus sizes with less than fifty clips (Ma et al., 2016; Zhang and Bull, 2019). In today's environment deployment at scale requires testing with larger corpus sizes. In addition the resolutions of content have increased drastically. Therefore we made a point in this work of generating results using a large corpus size of approximately 10,000 clips. We are able to show results at 720P and 1080P resolutions. It is also significant that we test on User Generated Content as opposed to pristine content. To our knowledge work done on this kind of variety and size of corpus is unique.

Overall, we have presented multiple systems to improve the compression of modern videos through per clip adjustments to the rate distortion optimisation of modern video codecs. By focusing on tailoring the codec on a per clip basis, and on a better representation of today's video, we achieve the bitrate savings presented in this thesis. It would make sense that this approach would not be only useful in the RDO algorithm, but other parts of the video codecs as well.

7.4 Future work

There is a lot of potential for further research. This work was just the beginning investigation into a deep research topic within the field of video compression. There are three possible areas of future work to expand on the research presented in this thesis. First is expanding the content and formats of the input videos to be compressed. Secondly, expanding the video codecs and the implementation of the direct optimiser within codecs and lastly, better quantifying the impact of this work through experiments on energy reduction.

7.4.1 Additional Video Content

It is worth exploring the idea of λ optimisation on High-Dynamic Range (HDR) material. HDR has been established in many market segments, especially capture devices, production and displays (Kunkel and Griffis, 2021). There is active research efforts to improve the use of HDR video throughout the different stages of video processing workflows, for example real-time content compositing (Kunkel and Griffis, 2021). HDR systems can capture, process, and reproduce a scene conveying the full range of perceptible shadow and highlight details beyond standard dynamic range (SDR) video systems. All clips used in this project have been SDR, and there is potential in expanding to HDR. We can see promise of this work in Vibhoothi et al. (2022a).

Similarly, the largest resolution used in the corpus is 1080p. With 4k, 8k and even larger super resolutions becoming more common in video production and content delivery systems, improving the compression of these clips may have a great impact for video content delivery systems. In addition, the current codec implementations might not be best suited for the newer video formats such as 360 degree video. It is worth investigating if re-engineering the RDO module within video codecs can provided additional improvement for the large resolutions, as well as 360 degree video content.

7.4.2 Additional Codec Options

This work focused on two codecs, HEVC and VP9. Expanding from these into the newer codecs, such as VVC and AV1 and beyond should lead to an additional impact. It was interesting that the initial work performed on using motion vectors from optical flow algorithms reached its full potential when moving from MPEG4 and H264 on to H265 and VP9 (details in Appendix 1). However, initial experiments using AV1 have indicated that there is still potential compression improvements to be found through the use of direct optimisation being applied to the Lagrangian Multiplier within codecs Vibhoothi et al. (2022a).

Throughout this work, the adjustments made to the RDO module were on a per clip basis. There is potential in expanding beyond this, to a per frame approach. In Chapter 4, we saw theoretical improvement from applying the direct optimisation techniques to the entire RD-curve to applying them on a per operating point level. It is expected that there will be additional improvement in moving to a per frame approach or even a granular per encoding macroblock/ coding tree unit. This will however, come at a higher computational cost. Initial experiments on per frame Lagrangian Multiplier Optimisation can be seen in Vibhoothi et al. (2022b).

7.4.3 Energy Consumption

There is a growing concern that the use of deep learning as well as applying multiple video encodes may lead to an energy consumption which far outweighs the potential savings achieved. Naturally, training, and implementing deep learning models comes at its own computational cost. A main limitation of this work is that we only give an estimate of the energy, through the metric of video encodes, used in this process compared to the direct optimiser. Using number of video encodes to compare the proxy or prediction systems to the direct optimisation system is an unfair comparison as it neglects any overhead required in computing the prediction as well the required energy to create proxies. It is feasible that for the deep learning systems presented here, there is not a significant reduction in energy used.

Further experiments to quantify the impact of this work in terms of energy used should be performed.

7.5 Final Remarks

Overall, this thesis demonstrates that there is significant value in per clip encoding. The tools which were developed for video encoding over time are beneficial, however they are limited in the performance as they were designed as a “one-size-fits-all” tool which is not necessarily suitable in today’s user generated video world. By tailoring the most prevalent tools in today’s internet video systems to the content which is being processed we are capable of providing better service for all through improved bitrates at equal quality in content delivery systems.

References

Anne Aaron, Zhi Li, Megha Manohara, Jan De Cock, and David Ronca. Per-title encode optimization. *The Netflix Techblog*, 2015. https://medium.com/netflix-techblog/per-title-encode-optimization-7e99442b62a2.

Mariana Afonso, Anush Moorthy, Liwei Guo, and Anne Aaron. Improving our video encodes for legacy devices, Aug 2020. URL https://netflixtechblog.com/improving-our-video-encodes-for-legacy-devices-2b6b56eec5c9.

Hadi Amirpour, Christian Timmerer, and Mohammad Ghanbari. Pstr: Per-title encoding using spatio-temporal resolutions. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2021.

Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *Int. J. Comput. Vision*, 92(1):1–31, March 2011. ISSN 0920-5691. doi: 10.1007/s11263-010-0390-2. URL http://dx.doi.org/10.1007/s11263-010-0390-2.

Abdelhak Bentaleb, Bayan Taani, Ali C Begen, Christian Timmerer, and Roger Zimmermann. A survey on bitrate adaptation schemes for streaming media over http. *IEEE Communications Surveys & Tutorials*, 21(1):562–585, 2018.

T Bernatin, G Sundari, A Sahaya Anselin Nisha, and MS Godwin Premi. Optimized inter prediction for h. 264 video codec. *International Journal of Electronics and Telecommunications*, 67(2):309–314, 2021.

G. Bjøntegaard. Calculation of average psnr differences between rd curves; vceg-m33. Technical report, 2001.

Netflix Technology Blog. Toward A Practical Perceptual Video Quality Metric, June 2016. URL https://medium.com/netflix-techblog/toward-a-practical-perceptual-video-quality-metric-653f208b9652.

Tim Borer and Thomas Davies. Dirac video compression using open technology. *BBC EBU technical review*, 303, 2005.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, Nov 2001. ISSN 0162-8828. doi: 10.1109/34.969114.

T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision (ECCV)*, volume 3024 of *Lecture Notes in Computer Science*, pages 25–36. Springer, May 2004. URL http://lmb.informatik.uni-freiburg.de/Publications/2004/Bro04a.

RECOMMENDATION ITU-R BT. Methodology for the subjective assessment of the quality of television pictures. 2002.

D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012.

J. C. Candy, M. M. A. Franke, B. G. Haskell, and F. W. Mounts. Transmitting television as clusters of frame-to-frame differences. *The Bell System Technical Journal*, 50(6): 1889–1917, July 1971. ISSN 0005-8580. doi: 10.1002/j.1538-7305.1971.tb02587.x.

Stephen Cass. The age of the zettabyte cisco: the future of internet traffic is video [dataflow]. *IEEE Spectrum*, 51(3):68–68, 2014. URL https://newsroom.cisco.com/press-release-content?type=webcontent&articleId=1853168.

Meixu Chen, Anjul Patney, and Alan C Bovik. Movi-codec: Deep video compression without motion. In *2021 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2021.

Yue Chen, Debargha Murherjee, Jingning Han, Adrian Grange, Yaowu Xu, Zoe Liu, Sarah Parker, Cheng Chen, Hui Su, Urvang Joshi, et al. An overview of core coding tools in the av1 video codec. In *2018 Picture Coding Symposium (PCS)*, pages 41–45. IEEE, 2018.

Yu M Chi, Trac D Tran, and Ralph Etienne-Cummings. Optical flow approximation of sub-pixel accurate block matching for video coding. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 1, pages I–1017. IEEE, 2007.

Marc Chriqui and Purnendu Sinha. Survey of motion estimation techniques for video compression. volume 4796, pages 4796–4796–9, 2003. doi: 10.1117/12.452142. URL https://doi.org/10.1117/12.452142.

Marcel Corrêa, Mário Saldanha, Alex Borges, Guilherme Corrêa, Daniel Palomino, Marcelo Porto, Bruno Zatt, and Luciano Agostini. Av1 and vvc video codecs: Overview on complexity reduction and hardware design. *IEEE Open Journal of Circuits and Systems*, 2:564–576, 2021.

Michele Covell, Martín Arjovsky, Yao-chung Lin, and Anil Kokaram. Optimizing transcoder quality targets using a neural network with an embedded bitrate model. *Electronic Imaging*, 2016(2):1–7, 2016.

Gerard de Haan and Paul W.A.C Biezen. Sub-pixel motion estimation with 3-d recursive search block-matching. *Signal Processing: Image Communication*, 6(3):229–239, 1994. ISSN 0923-5965. doi: https://doi.org/10.1016/0923-5965(94)90027-2. URL http://www.sciencedirect.com/science/article/pii/0923596594900272.

Hugh Everett III. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operations research*, 11(3):399–417, 1963.

Anja Feldmann, Oliver Gasser, Franziska Lichtblau, Eric Pujol, Igmar Poese, Christoph Dietzel, Daniel Wagner, Matthias Wichtlhuber, Juan Tapiador, Narseo Vallina-Rodriguez, et al. Implications of the covid-19 pandemic on the internet traffic. In *Broadband Coverage in Germany; 15th ITG-Symposium*, pages 1–5. VDE, 2021.

Chad Fogg, Didier J LeGall, Joan L Mitchell, and William B Pennebaker. *MPEG video compression standard*. Springer Science & Business Media, 2007.

Bernard Ghanem and Narendra Ahuja. Maximum margin distance learning for dynamic texture recognition. In *European Conference on Computer Vision*, pages 223–236. Springer, 2010.

Bernd Girod. Rate-constrained motion estimation. In *Visual Communications and Image Processing'94*, volume 2308, pages 1026–1035. International Society for Optics and Photonics, 1994.

Ahmed M Hamza, Abdelrahman Abdelazim, and Djamel Ait-Boudaoud. Parameter optimization in h. 265 rate-distortion by single frame semantic scene analysis. *Electronic Imaging*, 2019(11):262–1, 2019.

B. G. Haskell. Interframe coding of monochrome television - a review. volume 0087, pages 0087–0087–10, 1976. doi: 10.1117/12.954998. URL https://doi.org/10.1117/12.954998.

Jilei Hou. World's first software-based neural video decoder running hd format in real-time on a commercial smartphone [video], Jun 2021. URL https://www.qualcomm.com/news/onq/2021/06/17/worlds-first-software-based-neural-video-decoder-running-hd-format-real-time.

S. Im and K. Chan. Multi-lambda search for improved rate-distortion optimization of h.265/hevc. In *2015 10th International Conference on Information, Communications and Signal Processing (ICICS)*, pages 1–5, Dec 2015. doi: 10.1109/ICICS.2015.7459952.

J. Jain and A. Jain. Displacement measurement and its application in interframe image coding. *IEEE Transactions on Communications*, 29(12):1799–1808, Dec 1981. ISSN 0090-6778. doi: 10.1109/TCOM.1981.1094950.

Nuggehally S Jayant and Peter Noll. Digital coding of waveforms: principles and applications to speech and video. *Englewood Cliffs, NJ*, pages 115–251, 1984.

Y. Kameda, H. Kishi, T. Ishikawa, I. Matsuda, and S. Itoh. Multi-frame motion compensation using extrapolated frame by optical flow for lossless video coding. In *2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 300–304, Dec 2016a. doi: 10.1109/ISSPIT.2016.7886053.

Yusuke Kameda, Hiroyuki Kishi, Tomokazu Ishikawa, Ichiro Matsuda, and Susumu Itoh. Multi-frame motion compensation using extrapolated frame by optical flow for lossless video coding. In *Signal Processing and Information Technology (ISSPIT), 2016 IEEE International Symposium on*, pages 300–304. IEEE, 2016b.

Damian Karwowski, Tomasz Grajek, Krzysztof Klimaszewski, Olgierd Stankiewicz, Jakub Stankowski, and Krzysztof Wegner. 20 years of progress in video compression–from mpeg-1 to MPEG-H HEVC. general view on the path of video coding development. In *International Conference on Image Processing and Communications*, pages 3–15. Springer, 2016.

Ioannis Katsavounidis and Liwei Guo. Video codec comparison using the dynamic optimizer framework. In *Applications of Digital Image Processing XLI*, volume 10752, page 107520Q. International Society for Optics and Photonics, 2018.

Angeliki V Katsenou, Joel Sole, and David R Bull. Content-gnostic bitrate ladder prediction for adaptive video streaming. In *2019 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2019.

Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*, 2015.

Anil Kokaram, Thierry Foucu, and Yang Hu. A look into youtube's video file anatomy, Apr 2016. URL https://youtube-eng.googleblog.com/2016/04/a-look-into-youtubes-video-file-anatomy.html.

Janusz Konrad and Eric Dubois. Bayesian estimation of motion vector fields. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(9):910–927, September 1992. ISSN 0162-8828. doi: 10.1109/34.161350. URL https://doi.org/10.1109/34.161350.

Till Kroeger, Radu Timofte, Dengxin Dai, and Luc Van Gool. Fast optical flow using dense inverse search. In *European Conference on Computer Vision*, pages 471–488. Springer, 2016.

Shiba Kuanar, Christopher Conly, and KR Rao. Deep learning based hevc in-loop filtering for decoder quality enhancement. In *2018 Picture Coding Symposium (PCS)*, pages 164–168. IEEE, 2018.

Timo Kunkel and Patrick Griffis. 2021 high-dynamic range (hdr) progress report. *SMPTE Motion Imaging Journal*, 130(8):101–107, 2021. doi: 10.5594/JMI.2021.3095091.

Thorsten Laude, Yeremia Gunawan Adhisantoso, Jan Voges, Marco Munderloh, and Jörn Ostermann. A comprehensive video codec comparison. *APSIPA Transactions on Signal and Information Processing*, 8, 2019.

W. Li and E. Salari. Successive elimination algorithm for motion estimation. *IEEE Transactions on Image Processing*, 4(1):105–107, Jan 1995. ISSN 1057-7149. doi: 10.1109/83.350809.

Joe Yuchieh Lin, Lina Jin, Sudeng Hu, Ioannis Katsavounidis, Zhi Li, Anne Aaron, and C-C Jay Kuo. Experimental design and analysis of jnd test on coded image/video. In *Applications of digital image processing XXXVIII*, volume 9599, page 95990Z. International Society for Optics and Photonics, 2015.

S. Ling, Y. Baveye, P. L. Callet, J. Skinner, and I. Katsavounidis. Towards perceptually-optimized compression of user generated content (ugc): Prediction of ugc rate-distortion category. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2020.

Dong Liu, Zhenzhong Chen, Shan Liu, and Feng Wu. Deep learning-based technology in responses to the joint call for proposals on video compression with capability beyond hevc. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(5):1267–1280, 2019.

Dong Liu, Yue Li, Jianping Lin, Houqiang Li, and Feng Wu. Deep learning-based video coding: A review and a case study. *ACM Computing Surveys (CSUR)*, 53(1):1–35, 2020.

Xie Liyin, Su Xiuqin, and Zhang Shun. A review of motion estimation algorithms for video compression. In *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, volume 2, pages V2–446–V2–450, Oct 2010. doi: 10.1109/ICCASM.2010.5620542.

Guo Lu, Xiaoyun Zhang, Wanli Ouyang, Li Chen, Zhiyong Gao, and Dong Xu. An end-to-end learning framework for video compression. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3292–3308, 2020.

Chengyue Ma, Karam Naser, Vincent Ricordel, Patrick Le Callet, and Chunmei Qing. An adaptive lagrange multiplier determination method for dynamic texture in hevc. In *2016 IEEE International Conference on Consumer Electronics-China (ICCE-China)*, pages 1–4. IEEE, 2016.

Detlev Marpe, Heiko Schwarz, and Thomas Wiegand. Context-based adaptive binary arithmetic coding in the h. 264/avc video compression standard. *IEEE Transactions on circuits and systems for video technology*, 13(7):620–636, 2003.

Alexandre Mercat, Marko Viitanen, and Jarno Vanne. Uvg dataset: 50/120fps 4k sequences for video codec analysis and development. In *Proceedings of the 11th ACM Multimedia Systems Conference*, pages 297–302, 2020.

Loren Merritt and Rahul Vanam. Improved rate control and motion estimation for h. 264 encoder. In *2007 IEEE International Conference on Image Processing*, volume 5, pages V–309. IEEE, 2007.

Debargha Mukherjee, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje. The latest open-source video codec vp9-an overview and preliminary results. In *Picture Coding Symposium (PCS), 2013*, pages 390–393. IEEE, 2013.

A. N. Netravali and J. D. Robbins. Motion-compensated television coding: Part i. *The Bell System Technical Journal*, 58(3):631–670, March 1979. ISSN 0005-8580. doi: 10.1002/j.1538-7305.1979.tb02238.x.

VideoLAN Organization. x265, the free H.265/HEVC encoder, 2018. https://www.videolan.org/developers/x265.html.

Antonio Ortega and Kannan Ramchandran. Rate-distortion methods for image and video compression. *IEEE Signal processing magazine*, 15(6):23–50, 1998.

G. Ottaviano and P. Kohli. Compressible motion fields. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2251–2258, June 2013. doi: 10.1109/CVPR.2013.292.

Roger Pantos and William May. Http live streaming. Technical report, 2017. URL https://www.rfc-editor.org/rfc/rfc8216.

M. A. Papadopoulos, F. Zhang, D. Agrafiotis, and D. Bull. An adaptive qp offset determination method for hevc. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 4220–4224, Sep. 2016. doi: 10.1109/ICIP.2016.7533155.

William H Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery. Numerical recipes in c, 1988.

John G Proakis. *Digital Communications*. McGraw-Hill, 2000. ISBN 0-07-232-111-3.

The WebM Project. The WebM Project, VP9 Project Summary, 2019. https://www.webmproject.org/vp9/.

Iain E Richardson. *Video codec design: developing image and video compression systems*. John Wiley & Sons, 2002.

Iain E Richardson. *The H. 264 advanced video compression standard*. John Wiley & Sons, 2011.

Iain E Richardson. Video compression codecs. *International Association of Sound and Audiovisual Archives (IASA) Journal*, (47):8–21, 2017.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Sandvine. Global internet phenomena, 2021. URL https://www.sandvine.com/phenomena.

Shahid Mahmood Satti, Matthias Obermann, Roland Bitto, Christian Schmidmer, and Michael Keyhl. Low complexity" smart" per-scene video encoding. In *2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–3. IEEE, 2019.

Guido M Schuster and Aggelos K Katsaggelos. Fast and efficient mode and quantizer selection in the rate distortion sense for h. 263. In *Visual Communications and Image Processing'96*, volume 2727, pages 784–795. SPIE, 1996.

Y. Shen and C. Kuo. Two pass rate control for consistent quality based on down-sampling video in HEVC. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2018. doi: 10.1109/ICME.2018.8486544.

Robert Skupin, Christian Bartnik, Adam Wieckowski, Yago Sanchez, Benjamin Bross, Cornelius Hellge, and Thomas Schierl. Open gop resolution switching in http adaptive streaming with vvc. In *2021 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2021.

Iraj Sodagar. The mpeg-dash standard for multimedia streaming over the internet. *IEEE MultiMedia*, 18(4):62–67, 2011. doi: 10.1109/MMUL.2011.71.

C. Stiller and J. Konrad. Estimating motion in image sequences. *IEEE Signal Processing Magazine*, 16(4):70–91, Jul 1999. ISSN 1053-5888. doi: 10.1109/79.774934.

Christoph Stiller. Motion-estimation for coding of moving video at 8 kbit/s with gibbs-modeled vectorfield smoothing. volume 1360, pages 1360–1360–9, 1990. doi: 10.1117/12.24233. URL https://doi.org/10.1117/12.24233.

G. J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, Nov 1998. ISSN 1053-5888. doi: 10.1109/79.733497.

G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, Dec 2012. ISSN 1051-8215. doi: 10.1109/TCSVT.2012.2221191.

Jo Yew Tham, S. Ranganath, M. Ranganath, and A. A. Kassim. A novel unrestricted center-biased diamond search algorithm for block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(4):369–377, Aug 1998. ISSN 1051-8215. doi: 10.1109/76.709403.

Christian Timmerer, Mathias Wien, Lu Yu, and Amy Reibman. Special issue on open media compression: Overview, design criteria, and outlook on emerging standards. *Proceedings of the IEEE*, 109(9):1423–1434, 2021. doi: 10.1109/JPROC.2021.3098048.

Zhengzhong Tu, Chia-Ju Chen, Yilin Wang, Neil Birkbeck, Balu Adsumilli, and Alan C Bovik. Efficient user-generated video quality prediction. In *2021 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2021.

Vibhoothi, François Pitié, Angeliki Katsenou, Daniel Joseph Ringis, Yeping Su, Neil Birkbeck, Jessie Lin, Balu Adsumilli, and Anil Kokaram. Direct optimisation of λ for hdr content adaptive transcoding in av1. *arXiv preprint arXiv:2208.11150*, 2022a.

Vibhoothi, François Pitié, and Anil Kokaram. Frame-type sensitive rdo control for content-adaptive-encoding. *arXiv preprint arXiv:2206.11976*, 2022b.

Yilin Wang, Sasi Inguva, and Balu Adsumilli. Youtube ugc dataset for video compression research. *arXiv preprint arXiv:1904.06457*, 2019.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE International Conference on Computer Vision (ICCV)*, Sydney, Australia, Dec 2013. URL http://hal.inria.fr/hal-00873592.

T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan. Rate-constrained coder control and comparison of video coding standards. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):688–703, July 2003. ISSN 1051-8215. doi: 10.1109/TCSVT.2003.815168.

Ping-Hao Wu, Volodymyr Kondratenko, Gaurang Chaudhari, and Ioannis Katsavounidis. Encoding parameters prediction for convex hull video encoding. In *2021 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2021.

Xiph.org. Xiph.org Video Test Media, 2018. https://media.xiph.org/video/derf/.

Aisheng Yang, Huanqiang Zeng, Jing Chen, Jianqing Zhu, and Canhui Cai. Perceptual feature guided rate distortion optimization for high efficiency video coding. *Multidimensional Systems and Signal Processing*, 28(4):1249–1266, 2017.

S. I. Young and D. Taubman. Rate-distortion optimized optical flow estimation. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 1677–1681, Sept 2015a. doi: 10.1109/ICIP.2015.7351086.

Sean I Young and David Taubman. Rate-distortion optimized optical flow estimation. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 1677–1681. IEEE, 2015b.

F. Zhang and D. R. Bull. Rate-distortion optimization using adaptive lagrange multipliers. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1, 2019. ISSN 1051-8215. doi: 10.1109/TCSVT.2018.2873837.

Ce Zhu, Xiao Lin, and Lap-Pui Chau. Hexagon-based search pattern for fast block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(5): 349–355, May 2002. ISSN 1051-8215. doi: 10.1109/TCSVT.2002.1003474.

Appendix A

Using Modern Motion Estimators in Video Codecs

Motion estimation is a key component of any modern video codec. Our understanding of motion and the estimation of motion from video has come a very long way since 2000. A quick look at the current evaluation of optic flow techniques first presented in 2011 by Baker et al. (2011) and the corresponding online league table[1] show a comparison of 135 different algorithms. For frame interpolation accuracy, block based algorithms are at rank 41 at best. In the Sintel dataset for optic flow comparison (Butler et al., 2012) they show up as rank 137 out of 148 different algorithms. While this does not mean that better motion estimation techniques combined with rate control within a codec would have real impact, we believe its worth the experiment. Our contribution to the existing body of literature in motion estimation for encoding is to inform the ongoing debate of the impact of motion estimation performance on encoding performance, a debate ongoing since the 1990's.

[1] `http://vision.middlebury.edu/flow/`

A.1 Motion Estimation in Video Compression

The heart of all hybrid video compression schemes is a method for predicting pixels from spatial or temporal locations in the current or other frames. The idea of exploiting temporal redundancy for prediction dates back to early experiments by Candy et al. (1971); Haskell (1976). The image sequence model used for prediction has remained unchanged since then as follows.

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})) + e_n(\mathbf{x}) \tag{A.1}$$

In this model the pixel value at site $\mathbf{x}$ in frame n, $I_n(\mathbf{x})$, is predicted by the value of the pixel at location $\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})$ in a previous frame $n-1$, $I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}))$ allowing for some prediction error $e(\mathbf{x})$ at that site $\mathbf{x}$ in frame n. That error or residual is also called the motion compensated residual. We say that I_{n-1} is motion compensated with respect to frame n when it is shifted according to the motion between the frames $\mathbf{d}_{n,n-1}(\mathbf{x})$ at the relevant site $\mathbf{x}$. Data compression is achieved by transform coding of the residual $e_n(\cdot)$.

This model is well known to fail in occluded/uncovered areas and so the prediction equation can be extended to support from the next as well as previous frames. This bi-directional prediction mode was introduced quite early on in the development of video codecs and incorporated into MPEG2,4 H.264. The current evolution of compression standards with VP9, H.265, H.266 and AV1 have extended the idea further to allow prediction from any other kind of supporting frame, be it in the sequence or auxiliary to it.

To generate a solution for the motion vector $\mathbf{d}_{n,n-1}$ in video compression the key assumption has been that the motion field is piecewise constant. Hence motion is constant over a block of a size that varies from 2×2 pixels up to 128×128 pixels in recent standards. The codec adaptively chooses both that block size and the motion vector associated with that block. This class of techniques is known as Block Matching. Efficient search algorithms have been explored since 1979 when gradient based search was first introduced by Netravali and Robbins (1979). Since then, the estimation of $\mathbf{d}_{n,n-1}$ has been dominated by direct search

techniques for solving equation A.1 e.g. Logarithmic search (Jain and Jain, 1981), Diamond Search (Tham et al., 1998), Sucessive Elimination (Li and Salari, 1995), 3DBM (de Haan and Biezen, 1994) and Hexagon Search Pattern (Zhu et al., 2002). Many of these are summarised in Chriqui and Sinha (2003); Liyin et al. (2010).

The optimisation cost function in block matching for choosing the best motion vector has traditionally been some variant of the sum of the absolute or squared residuals/prediction error over a block, SAE or SSE. For video compression, Girod (1994) first presented rate distortion criteria appropriate for motion estimation in a codec. That has since become part of the optimisation scheme used in all codecs today (Sullivan and Wiegand, 1998). The essential idea is to choose a motion vector which minimises the prediction energy as well as the information needed to encode the motion vector itself. This motion estimation rate distortion energy $E(\mathbf{d})$ is stated as follows.

$$E(\mathbf{d}) = \lambda R(\mathbf{d}) + \sum_{\mathbf{x} \in B} |e_n(\mathbf{x}, \mathbf{d})| \tag{A.2}$$

where B indicates a block, $|e_n(\mathbf{x}, \mathbf{d})|$ is the SAE (for instance), $R(\mathbf{d})$ is the rate or bits required to encode the vector $\mathbf{d}$ and λ is a hyperparameter which controls the importance of rate control versus distortion management.

A.1.1 Modern motion estimation and video compression

Motion estimation since 2000 has been characterised by the quantitative application of temporal and spatial smoothness constraints. This was well articulated with the Bayesian approach by Konrad and Dubois (1992); Stiller and Konrad (1999). Other landmark work followed using optimisation strategies like Graph Cuts (Boykov et al., 2001) and variational methods together with second order constraints on the motion field (Brox et al., 2004). Much of this development is tracked by the survey work of Baker et al. (2011). A handful of authors have reported on the use of optic flow or modern motion estimation in codecs.

Stiller (1990) employed spatial smoothness constraints in block matching motion estimation for a codec at 8Kbits/sec. He reported about 1dB improvement at the same bitrate over standard block matching for three short sequences *Miss America, Swing, Alexis* which was state of the art at the time. It was not clear what codec was being used then.

Ottaviano and Kohli (2013) point out that rate distortion criteria can and should be integrated within the new optical flow algorithms for their use in video compression. They develop a wavelet based motion estimation algorithm using the RD criterion above, and apply it to a variant of the Dirac codec (Borer and Davies, 2005). They report gains from 1-49% on bitrate at the same PSNR when compared to the Dirac codec with block matching.

Young and Taubman (2015a) presented a more elaborate RD scheme for motion vector estimation in a codec and deployed variational methods for optimisation. Like Ottaviano and Kohli (2013) they deploy a wavelet model of the motion field. They report about 3dB improvement at the same bitrate for 4 CIF sequences *flower, foreman, mobile, harbour*. These gains were modeled based on measured motion compensated frame differences and not from an actual codec implementation.

Other work that employed modern optic flow approaches (Kameda et al., 2016a) concentrated on motion compensation using dense flow to extrapolate a new frame for prediction, rather than using the decoded frame. Gains in bitrate are reported at about 7% when motion is steady across frames, and negative when there is unsteadiness in the shot. This makes sense since extrapolation is unlikely to be successful when temporal motion is not smooth.

The motion estimation module of a video codec can have a varying performance based on the content of the video sequence. Since the content of a video sequence has an impact on the encoder performance, it is sensible to investigate if the video encoding process can become content adaptive.

A.1.2 Scope & Research Question

Entire codecs are convoluted and complex. The creation of a new codec or the development of a video codec is not within the scope of this research. The most widespread used video codecs are VP9 and H.265 (also known as HEVC) , as well as predecessors H.264 and MPEG4. Newer codecs such as AV1 and VVC (H.266) are not in widespread use at the moment. This work will consider the modification of MPEG4, H.264, H.265 and VP9 to ultimately maximize performance given a specific video. In particular, the motion estimation and prediction module can possibly lead to improved performance if adjusted based on the content of a video sequence.

This produces the research question for the work:

Can the parameters of the sub-modules of a video codec be automatically adjusted to ultimately maximize performance for a particular video ?

In order to answer the research question, certain experiments needed to be performed. These experiments could be categorized as “Motion Vector Insertion". Block diagrams giving a brief summary of the parameters of each experiment are found in Figures A.2, A.4 and A.7

A.2 Results

All work we have done has been performed on four of the most widespread video codecs: MPEG4, H.264, H.265 and VP9. The first experiments conducted focused on the insertion of ground truth motion vectors into the video codecs MPEG4 and H.264. The SINTEL sequence set (Butler et al., 2012) was used for evaluation of performance with ground truth motion. This dataset was chosen as the sequences are some of the longest with available ground truth motion data (Butler et al., 2012). Each of the sequences was 50 frames long with resolution 1024x436 and YUV420 pixel format. Sample frames of this dataset can be seen in Figure A.1. The SINTEL dataset can be split into two categories, albedo and final. Albedo frames, which are flat, unshaded, surfaces exhibit constant albedo (light absorption)

over time are seen on the left. Final frames which are full rendering with all effects including blur due to camera depth of field, motion and atmospheric effects (Butler et al., 2012).

Fig. A.1 *Sample frames from the SINTEL dataset. Albedo frames, which are flat, unshaded, surfaces exhibit constant albedo over time are seen on the left. Final frames which are full rendering with all effects including blur due to camera depth of field, motion and atmospheric effects (Butler et al., 2012)*

A.2.1 Down-sampling Optical Flow Vectors for use in Video Codecs

In order to use the pixel dense motion vectors in a block based motion estimation scheme, some form of downsampling must happen. Three approaches were taken, Mean, Vector Median and the "Best Dense Available". Within a size block, Mean refers to the average of the vectors within that block would be the representative motion vector for use in the codec. Vector Median refers to the spatial median of the pixel dense motion vectors in the block to be inserted to the codec. For the "best dense available" scheme, each of the motion vectors in the $m \times n$ block were inserted into the RD-equation and whichever provided the lowest cost was selected as the representative motion vector for use in the video encoder.

Another method which was deployed was the “Fused" system, this compared the RD-cost of the block using both the internal block matcher’s motion vector to the RD-cost of the block using the representative motion vector from the dense motion vectors. Whichever motion vector provided the lower cost, was used in the codec. Fused-Mean, Fused-Median and Fused-BestDense refer to comparing the block matchers motion vector’s RD cost to the RD-cost for the representative motion vector generated by the Mean, Vector Median and Best Dense Available methods respectively.

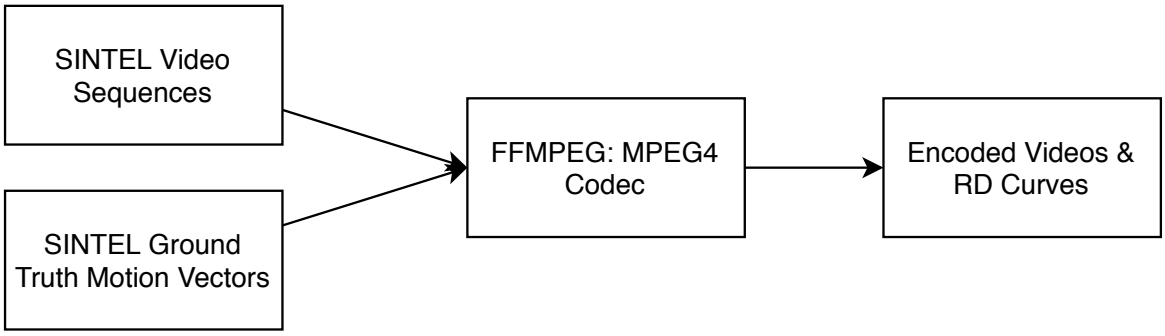

Fig. A.2 *Experiment: Using SINTEL sequences and Ground Truth with the MPEG4 Codec. The ground truth motion vectors were downsampled using either vector median or mean of the vectors within a macroblock*

The first experiment pertaining to motion vector insertion, used the MPEG4 codec. The implementation of this codec was using the open source library FFMPEG. FFMPEG is the “leading multimedia framework" for video encoding and decoding. The motion estimation sub-component was modified in order to use the motion vectors provided by SINTEL. The videos were encoded over an exhaustive range of constant quantiser settings, and BD-Rate and BD-PSNR were calculated comparing the default block matcher’s encoding results to each of the proposed methods of motion vector insertion. This experiment resulted in the following output curves:

- **Diamond** - The direct search block matcher; This is the results of the video encoder with no modifications done.
- **Zero Motion** - This is the result when no motion information is provided to the encoder

- **Mean** - Using the ground truth motion vectors. In order to downsample the dense motion vector field for use in a block based system, the mean of the motion vectors was used.

- **Median** - Using the ground truth motion vectors. In order to downsample the dense motion vector field for use in a block based system, the vector of the motion vectors was used.

- **Hybrid-Median** - This used both the diamond and median results. For each macroblock in the frame, the bit cost of both the motion vector from the direct search block matcher and the vector median of the ground truth motion vectors were calculated. Whichever motion vector resulted in a lower cost was used.

Figure A.3 shows the different rate distortion curves for the Sintel sequences using MPEG4 and the ground truth provided by Sintel. The first observation is that when using no motion compensation (Zero Motion), the codec performs significantly worse than the default settings. It was also interesting to note, that using the ground truth motion alone in the codec provided video files which were larger and of lower quality than those which were coded using the motion vectors generated by the direct block search. On a positive note, using the fused or hybrid methods, a 1.6% BD-Rate improvement was seen across the dataset.

As with the MPEG4 experiment, the SINTEL sequences were used to determine if there is any improvement when using the dense motion vectors. In addition to using the ground truth, videos were also encoded using an external motion estimator on the decoded frames. The ground truth motion vectors were labeled **T0** and motion vectors generated using an external motion estimator on the decoded frames were labeled **T2**.The motion estimators used were DisFast (Kroeger et al., 2016) and DeepFlow (Weinzaepfel et al., 2013).

Results of this experiment can be seen in figure A.5. Hybrid or Fused methods outperform the other methods of generating motion vectors for video compression. With up to 4.6% BD-Rate improvement for a single sequence and 1.6% BD-Rate improvement across the

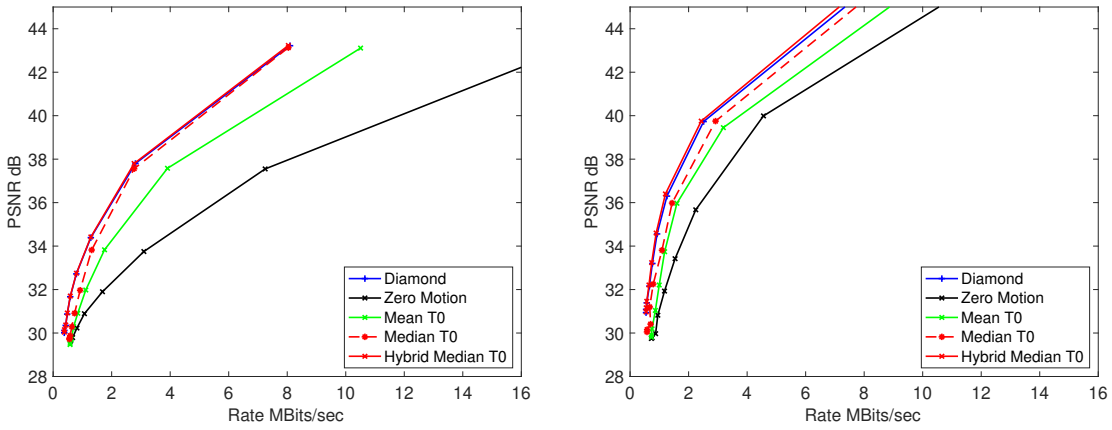

Fig. A.3 *Aggregated RD Curves for 5 different motion estimator types compared across the Sintel Dataset using MP4. Left: Albedo render Right: Final render. The Albedo rendered scenes tend to be more cartoonish than the final renders. The final renders contain more motion blur, lighting finishes, shadows etc. That means the performance of the codec is generally better with the final renders than the albedo renders.*

dataset. There was no advantage of using the ground truth over motion vectors generated by one of the motion estimators. This allowed for experimentation to be done on other video sequences which ground truth is not always available.

A.2.2 Expansion to newer codecs and live video sequences

Given the encouraging results of using external motion vectors in the older video codecs, further investigation was performed using the most common video codecs, H265 and VP9. Using the software implementations for H265 (Organization, 2018) and VP9 (Project, 2019), experiments were conducted on both the SINTEL sequences and additional live action sequences taken from Xiph.org (2018). This is a well known repository of publicly available raw video sequences and have been used in many video compression experiments. Sample frames from this dataset can be seen in Figure A.6.

Unfortunately, the improvements were not seen in the newer codecs, H.265 (HEVC) and VP9. Figures A.8 and A.9 show that the fused/hybrid methods only slightly outperform the default block matcher by more than 0.3% BD-Rate improvements for each codec across the SINTEL dataset. This small improvement was also seen for the live action sequences.

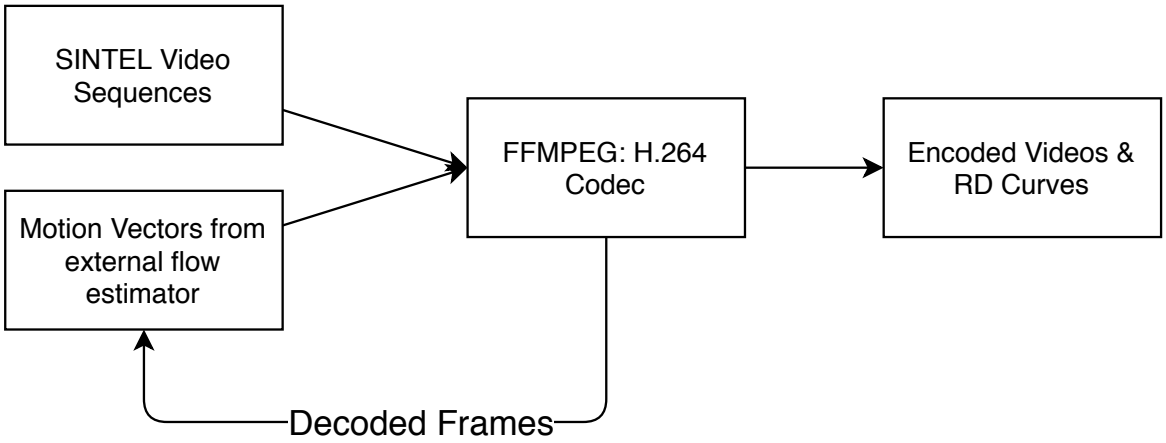

Fig. A.4 *Experiment: Using SINTEL Sequences and the motion vectors generated by an external motion estimator using the decoded frames as references. As with the previous experiment, the generated dense motion vectors were downsampled for use in the block based codec*

The percentage of blocks which use the dense motion vector as compared to using the motion vector generated by the direct search built into the codecs is approximately 1%. This 1% of the blocks is where the improvement from inserting dense motion vectors comes from. Further investigation on these blocks needs to be conducted, in order to determine if the dense motion estimator should be called upon in specific situations and areas of images.

A.3 Discussion

In general, these results encourage further thought. When used inside a codec with rate control, gains with the use of an optic flow estimator seem to disappear. Rate gains seem to be about 1-4% depending on the sequence and the codec, while PSNR gains are modest at about 0.1dB. In many ways, the results do not reflect the promise of previous work which report gains of the order of 1dB or as much as 49% in rate.

Figure A.10 shows a comparison between the motion fields estimated by the *Diamond* search block matcher in ffmpeg, Dis-Fast (T1), DeepFlow (T1) and ground truth (T0). We see that DeepFlow is indeed best as far as ground truth is concerned, with Dis-Fast second best. In some sequences (Bamboo and Temple) though, they are about as different from

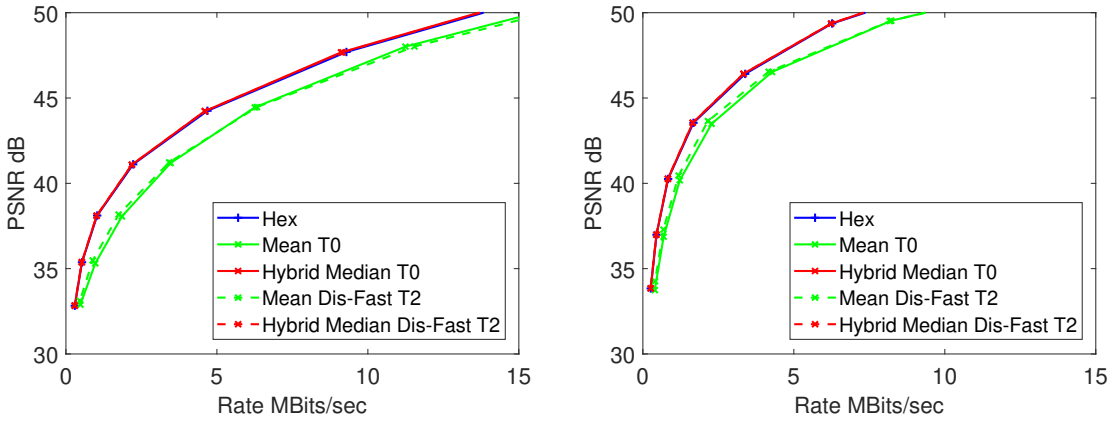

Fig. A.5 *Aggregated RD Curves for 3 different motion estimator types compared across the Sintel Dataset using x264. Left: Albedo render Right: Final render. The performance of the codec is generally better with the final renders than the albedo renders.*

ground truth as Diamond. The bottom row also shows that for those sequences they are also different from each other. Those sequences show complicated/fast motion and textures scenes. This might imply that Dis-Fast and Deep Flow are just not that different enough to generate much better performance than *Diamond* inside a codec for those scenes. Those are the scenes with very small BD gains.

The optic flow algorithm here was incorporated into the codec in a rather naive fashion, ignoring RD motion criterion. The fact that the Hybrid estimators outperformed Diamond (albeit by a small margin) shows that using RD criterion in the optic flow optimiser may be the best approach to increase the gains.

Finally, this work considered P frames only, and in a long GOP size (100 frames). That means that decoded frames can expect to deviate significantly from the I frame by the end of the GOP. Because we use only P-frames, that means that the direct search used in the codec may have a better chance of reducing prediction error than the implicit search with smoothness constraints of the optic flow motion estimator.

We shall continue to develop this experimental work in the future. It is sensible to consider next, motion estimators higher up in the rank than Dis-Fast or DeepFlow, a more

Fig. A.6 *Sample Frames for live action sequences. All were taken from the Derf's media repository (Xiph.org, 2018)*

useful parameter set for these algorithms (they were deployed out of the box here); and the impact of the use of B-frames for resolving occlusion/uncovering.

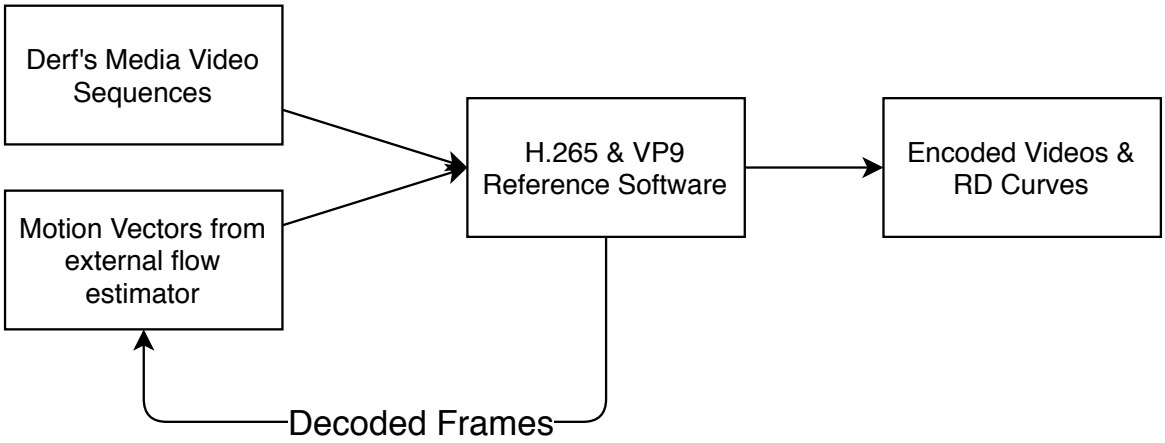

Fig. A.7 *Experiment: Using the newer video codecs for the motion vector insertion experiments. Motion Vectors were generated using the decoded frames with the deep flow motion estimator. downsampling was done using the vector median or best-dense methods*

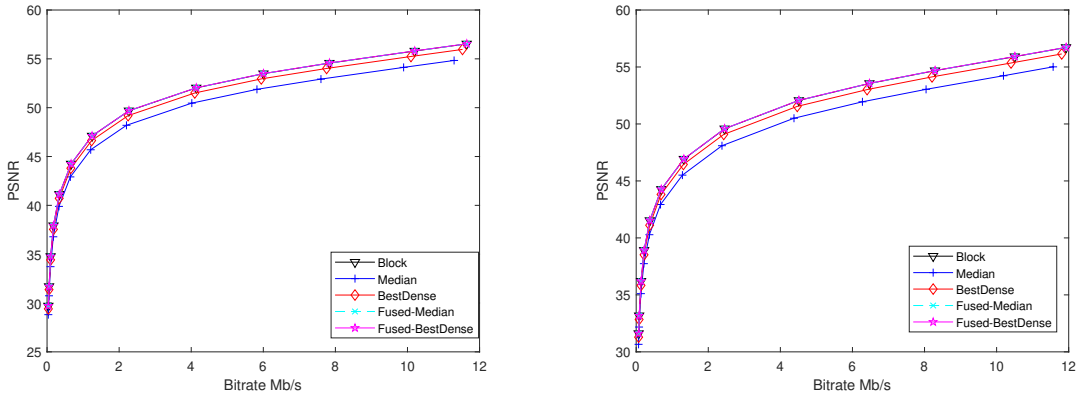

Fig. A.8 *Aggregated RD-Curves for the live action sequences (720p) encoded using different motion estimators for VP9 (left) and H.265 (right). If the insertion of the optical flow vectors led to improvements over Block Matching, the RD-Curves of* `Median`*,* `Fused-Median`*,* `BestDense` *and* `Fused-BestDense` *systems would be closer to the top left of the graph compared to* `Block`*. However we can see that the block matcher performed as good as the optical flow vectors with very little difference between the RD-Curves for both codecs. On average we measured a 0.3% improvement.*

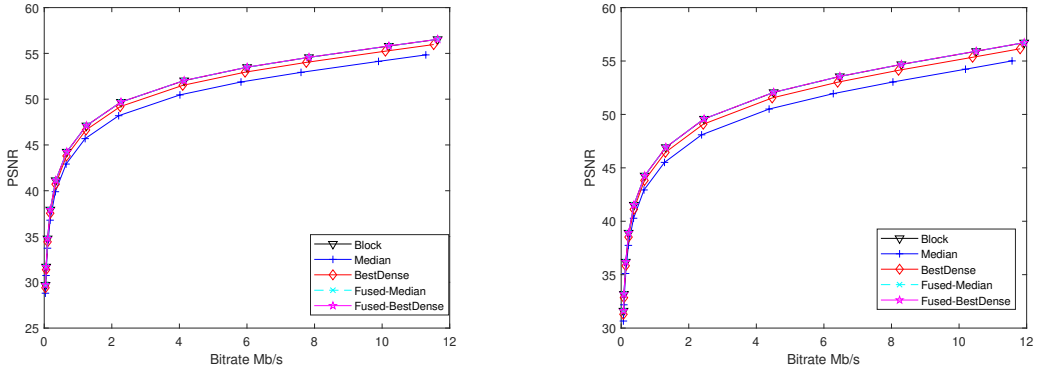

Fig. A.9 *VP9 and H265 Results for live action sequences*

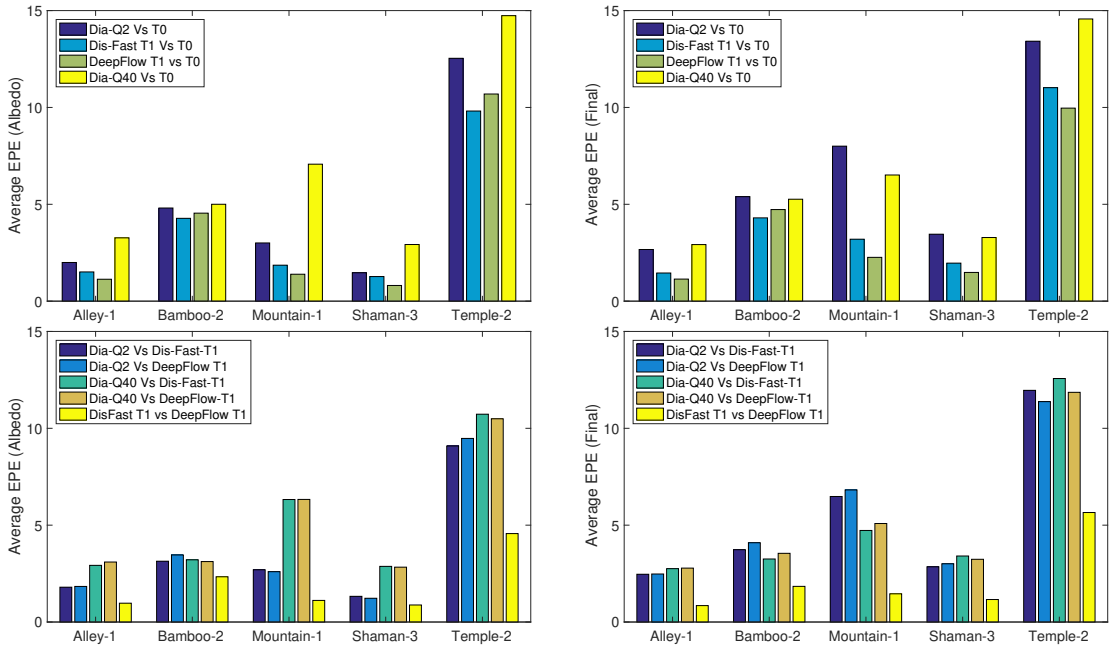

Fig. A.10 *Showing End Point Error for 5 Sintel sequences using 3 different motion types compared to ground truth (T0) and each other. Top Row : differences with ground truth, Bottom Row : differences between each other. DeepFlow yields the closest result to ground truth. At a quantiser value of 40 Diamond deviates from T0 much more than at* $q = 2$*. That is sensible because* $q = 40$ *implies a much lower picture quality and hence the estimated motion vectors would not necessarily be well matched to ground truth. The differences in the bottom row are an indicator that there is diversity in the motion estimates.*

Appendix B

Dataset and Demonstrations

Full details of the datasets used for the experiments and video demonstrations of the improved compression performance through the systems detailed in this thesis are available at the following site:

https://ringisd.github.io/Appendix2/

B.1 Initial Corpus

The clips used from Wang et al. (2019) in our initial corpus can be found in Table B.1.

B.2 Encoding Features

The Feature-set used can be found in Table B.2.

Table B.1 *List of 71 clips from Wang et al. (2019) used in preliminary experiments.*

Animation_360P-08c9	Gaming_720P-6658	NewsClip_360P-0376
Animation_360P-47cc	HowTo_360P-0562	NewsClip_360P-1eae
Animation_360P-4edc	HowTo_360P-2fd5	NewsClip_360P-4288
Animation_720P-1a6d	HowTo_360P-7dcd	NewsClip_720P-35d9
Animation_720P-4268	HowTo_720P-0c47	NewsClip_720P-7345
Animation_720P-4b17	HowTo_720P-21c6	NewsClip_720P-7745
Animation_720P-6372	HowTo_720P-6791	Sports_360P-2ace
CoverSong_360P-1d11	HowTo_720P-7878	Sports_360P-301d
CoverSong_360P-53a6	LiveMusic_360P-2508	Sports_360P-6f62
CoverSong_360P-5d20	LiveMusic_360P-54d0	Sports_720P-00a1
CoverSong_720P-3dca	LiveMusic_360P-6a65	Sports_720P-0b9e
CoverSong_720P-449f	LiveMusic_720P-2620	Vlog_360P-2e9d
CoverSong_720P-7360	LiveMusic_720P-4ae2	Vlog_360P-4795
CoverSong_720P-7539	LiveMusic_720P-6452	Vlog_360P-7efe
Gaming_360P-1e31	LyricVideo_360P-11eb	Vlog_720P-03d5
Gaming_360P-5e0f	LyricVideo_360P-17ce	Vlog_720P-11c5
Gaming_360P-73c7	LyricVideo_360P-3afc	Vlog_720P-5364
Gaming_720P-5ba2	LyricVideo_720P-4253	Sports_720P-3ffe
Gaming_720P-6403	LyricVideo_720P-739a	Sports_720P-6bb7
Gaming_720P-6625	MusicVideo_360P-17e4	TelevisionClip_360P-29f1
MusicVideo_360P-5358	TelevisionClip_360P-7b23	TelevisionClip_360P-74dd
MusicVideo_360P-5f07	TelevisionClip_720P-02b8	
MusicVideo_720P-0355	TelevisionClip_720P-1b61	
MusicVideo_720P-3285	TelevisionClip_720P-2b20	
MusicVideo_720P-4895	TelevisionClip_720P-5e93	

Table B.2 *List of Encoding Features.*

Class	bitrate *Y/((U+V)/2)
Original WIDTH	bitrate *P/B Ratio Y
Original HEIGHT	bitrate *P/B Ratio U
rescaled W	bitrate *P/B Ratio V
rescaled H	bitrate *P/B Count
bitrate	bitrate *P/B Size
YPSNR	Y/((U+V)/2) *P/B Ratio Y
UPSNR	Y/((U+V)/2) *P/B Ratio U
VPSNR	Y/((U+V)/2) *P/B Ratio V
P count	Y/((U+V)/2) *P/B Count
P ave-QP	Y/((U+V)/2) *P/B Size
P kbps	P/B Ratio Y *P/B Ratio U
P-PSNR Y	P/B Ratio Y *P/B Ratio V
P-PSNR U	P/B Ratio Y *P/B Count
P-PSNR V	P/B Ratio Y *P/B Size
B count	P/B Ratio U *P/B Ratio V
B ave-QP	P/B Ratio U *P/B Count
B kbps	P/B Ratio U *P/B Size
B-PSNR Y	P/B Ratio V *P/B Count
B-PSNR U	P/B Ratio V *P/B Size
B-PSNR V	P/B Count *P/B Size
Y/((U+V)/2)	CHUNK_COMPLEXITY_VARIATION
P/B Ratio Y	
P/B Ratio U	
P/B Ratio V	
P/B Count	
P/B Size	