SUPER-RESOLUTION AND SCALABLE VIDEO CODING

A Thesis

by

YIXU CHEN

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Xiong, Zixiang
Committee Members,	Wang, Zhangyang
	Hou, I-Hong
	Kalafatis, Stavros
Head of Department,	Dr. Aniruddha Datta

May 2021

Major Subject: Computer Engineering

ABSTRACT

Bits on the wire not only impact video quality delivered to customers but also drive the costs of video streaming services. This project aims at building state-of-the-art deep-learning-based video super-resolution (VSR) algorithms while addressing the compression artifact, and then integrate the VSR into the Scalable Extension of High Efficiency Video Coding (SHVC) by replacing the inter-layer upscaler with the VSR upscaler, and benchmark the codec performance. The discrete cosine transform upsampling filter in SHVC is applied to the base layer reconstructed video, therefore the reference video for the enhancement layer has scaling artifacts and compression artifacts. The VSR model used to perform upsampling can provide a higher quality reference for the EL.

However, the traditional VSR model can't be directly used on the SHVC as the low-resolution video used as input for the VSR upscaler is pristine but in SHVC the LR input to the upscaler is compressed by the base layer codec. High-frequency details are lost during the compression and artifacts are introduced by the block-based hybrid video coding framework. Therefore the video super-resolution models need to be modified as the compression process added artifacts to the input. In this work deartifact network (DANet) was introduced to perform the artifacts reduction and super-resolution at the same time. DANet is based on FRVSR which estimates the optical flow between frames and uses motion compensation to align local frames for the super-resolution.

Our DANet performs 0.28dB and 0.81 VMAF better than the FRVSR on the PRIME7 test set with CRF23 LRC video as input.

After integrating DANet to the SHVC codec, on the PRIMT7 test set, using PSNR as the metric, this VSR-integrated scalable video coding framework achieved -5.62% BD-rate reduction at the same video quality and 0.17 dB BD-PSNR quality improvement at the same bitrates compared with the original SHVC. Using VMAF as the metric, our VSR-SHVC achieved -10.01% BD-rate reduction and 0.79 BD-VMAF quality improvement.

DEDICATION

To my parents.

ACKNOWLEDGMENTS

This project is funded by Amazon Prime Video. I would like to thank my advisor, Professor Zixiang Xiong, for the guidance and valuable advice. I'd like to acknowledge the effort of my managers Sriram Sethuraman, Yongjun Wu and Hai Wei for their practical suggestions and invaluable insights. Special thanks to my committee members professor Zhangyang Wang, professor I-Hong Hou and professor Stavros Kalafatis.

NOMENCLATURE

HEVC	High Efficiency Video Coding
X265	A H.265 / HEVC video encoder application library
SHVC	Scalable Extention of HEVC
SHM	The name of the SHVC reference software
BL	Base layer of SHVC
EL	Enhancement layer of SHVC
ILR	Inter-layer reference pictures in SHVC
DCTIF	Discrete cosine transform interpolation filters
VSR	Video super-resolution
HR	High resolution video, ground truth of VSR model
LR	Low resolution video, input of VSR model
LRP	Pristine LR, the downsampled HR without compression
LRC	Compressed LR, the reconstructed LRP video
TMVP	Temporal motion vector prediction
QP	Quantization parameter
CRF	Constant Rate Factor, an encoder rate control method
CBR	Constant Bitrate, an encoder rate control method
TID	Temporal layer ID in SHVC
POC	Picture order count, the display index
GOP	Group of pictures

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION AND RELATED WORK

This work is composed of two parts, video super-resolution and scalable video codec. The super-resolution methods that this work is based on are introduced in 1.1. The scalable video codec framework SHVC and the original resampling process DCTIF are introduced in 1.2.

1.1 Super Resolution

The super-resolution algorithm aims to recovery high-frequency detail from the given low-resolution input. Super-Resolution is an ill-posed under-determined problem as there are infinite solutions for the same input. Therefore the prior knowledge is necessary to generate a reasonable solution especially when the upscaling factor is large and the structural information is completely lost during downsampling.

Based on whether to use temporal information between frames there are two different categories: single image super-resolution and video super-resolution.

1.1.1 Single Image Super-Resolution

A single image super-resolution (SISR) algorithm can construct a high-resolution image (HR) from a low-resolution input (LR) with only a single frame. Temporal consistency can't be guaranteed when applying SISR on videos as no temporal information was used when generating a SR frame. A few related SISR methods are introduced in this section then we will focus on video super-resolution in our work.

1.1.1.1 CNN-based approaches

Started with SRCNN[1] in 2014, convolutional neural networks are used on super-resolution tasks. SRCNN used only 3 layers of CNN while VDSR[2] used 20 layers and residual learning to achieve better results. Both of them are **pre-upsampling methods** using the interpolated image of HR size as input. The models only enhance the input image without upsampling. Therefore, the same training can deal with multiple scales. As shown in VDSR[2], the training data mixed

with all scaling factors can even boost the performance. However, the pre-upsampling methods are time-consuming as all calculations are done in the HR resolution.

EDSR[3] using the **post-upsampling method** which puts the upsampling at the end of the network to reduce the computation. The post-upsampling method usually can't deal with multiple scaling factors in a single model as in the pre-upsampling method. However, EDSR[3] trained a model called MDSR that fit multiple scales by sharing the main branch of the model and selecting the scale-specific upsampling layer. Residual blocks similar to ResNet but without batch normalization are used in this structure. And L1 loss instead of L2 loss is used to train the network.

1.1.1.2 GAN and perceptual loss

The GAN and perceptual loss are also used in training the SISR network. SRGAN[4] and EnhanceNet[5] used perceptual loss form pre-trained VGG network[6] and adversarial training. These models can synthesis texture that is completely lost in the downsampling process but the PSNR and SSIM can't capture the subjective performance of those models. Even though the PSNR or SSIM result is lower than other models, the output is more photo-realistic. However, in our case, the hallucinated finer detail is unnecessary in video coding which might incur even larger costs when calculating the residual between the VSR output and the ground truth.

1.1.1.3 Filter-based approach

RAISR[7] and DUF[8] both try to learn the upsampling filter, although DUF[8] is a VSR algorithm. RAISR[7] generated the filters by solving the least-squares minimization problem for each cluster based on the statistics of the local gradient. This method chooses from a cluster of fixed filters based on the gradient of the local patch. The filter used is fixed after training but using local gradient statistics as the hash table's key making it efficient.

In this work, an implementation of RAISR [7] are used as the benchmark against VSR methods.

1.1.2 Video Super-Resolution

For video super-resolution (VSR), it is impractical to simply apply single image super-resolution on each frame, as videos have inter-frame temporal dependency which needs to be fully exploited

to have an temporal consistent output. Depending on how to utilize temporal local information VSR models can be categorized into sliding windows structure and recurrent structure. Many VSR models also follow the alignment-fusion paradigm.

Sliding Window

The earlier study focused on the sliding window approach which takes multiple consecutive LR frames as input and utilizes local information to construct a single HR frame. Among those methods, the alignment-fusion structures are most popular such as TDAN[9], EDVR[10]. There are also filter-based approaches like DUF[8]. However, the sliding window method is time-consuming, as multiple frames are processed to generate one output and the same input LR frame can be processed several times. And the information used to generate the current output is limited by the size of the window.

Alignment-fusion

TDAN[9] and EDVR[10] both follow the alignment-fusion framework and use 5 frames as the window size. In the alignment-fusion framework, local frames are first aligned with the current frame by optical flow motion compensation or deformable convolution. Then a fusion module is designed to fuse the aligned local frames into one SR output. The alignment-fusion architecture can follow the recurrent structure like FRVSR[11] or sliding window methods like TDAN[9], EDVR[10]. The drawback of this framework is that inaccurate alignment will lead to worse SR performance.

Reucrrent structure

The recurrent structure is different from the sliding window approach which has the potential to propagate information about the scene unlimitedly like FRVSR[11] and RLSP[12].

The recurrent structure explicitly or implicitly keeps information flow to make use of the inter-frame correlation. With the one-input-one-output structure, these models have less calculation than the sliding window methods. However, the frames have to be input by the display order which makes it difficult to be integrated into random access video codec where the encoding order is different from the display order.

GAN

There are also GAN-based approaches like TecoGAN[13] which have the best subjective quality but the texture synthesized by those models are usually different than the ground truth making it inappropriate for the video codec to reference. Therefore, we won't use the GAN-based models in our experiment.

The following models are investigated: DUF[8], TDAN[9], EDVR[10], FRVSR[11], RLSP[12]. The comparision is in Table 1.1.

DUF is a sliding-window, filter-based VSR model and is introduced in 1.1.2.1. TDAN and EDVR are sliding-window, alignment-fusion VSR model and are introduced in 1.1.2.2 and 1.1.2.3. FRVSR and RLSP are recurrent VSR model and are introduced in 1.1.2.4 and 1.1.2.5.

Methods	Sliding Window	Alignment-Fusion	Loss function
DUF	7 frames	No. Dynamic filter generation	Huber loss
TDAN	5 frames	Yes. Deformable convolution	L1. Alignment loss and SR loss
EDVR	5 frames	Yes. Pyramid cascading deformable convolution	L2. Charbonnier penalty function
FRVSR	Recurrent	Yes. Optical flow and motion compensation	L2. Alignment loss and SR loss
RLSP	Recurrent	No explicit alignment	L2. SR loss

Table 1.1: VSR methods summarization

1.1.2.1 DUF

DUF[8] is a filter-based approach that uses 7 frames as the window size. The filter is generated by the filter generation network taking the motions between local frames into account. Then the filter will be convolved with the current frame and added to the residual generated by the residual generation network to get the final output. Huber loss is used to train the network.

1.1.2.2 TDAN

Deformable convolution first proposed in [14] is used in many high-level vision tasks like object detection and segmentation etc. TDAN[9] first uses deformable convolutions in video SR to align the feature map of the local frames. Convolution layers are first applied to local frames to extract features. Then the deformable convolution is applied to align the feature maps. The temporal alignment is done in the feature space and another convolution layer is added to reconstructed the aligned frame. Then all the aligned frames inside the window are concatenated with the current frame and feed into the SRNet. The current frame is used as the ground truth to calculate alignment loss. Similar to FRVSR, the final loss is the summation of alignment loss and SR loss. The benefit of deformable convolution is the sampling grid of convolution is learned by the network therefore the same CNN layer can have different receptive fields to better deal with objects of different scales and deformation. TDAN uses deformable convolution to avoid explicit motion compensation. Instead of pixel position mapping in optical flow, the alignment in the feature map can achieve better accuracy. This model still has the common problem of the alignment-fusion architecture that misalignment and unalignment will adversely affect the performance of the fusion module.

1.1.2.3 EDVR

Inspired by TDAN[9] and methodology in optical flow, EDVR[10] uses Pyramid processing, Cascading refinement and Deformable convolutions (PCD) module to align local frames. In addition to deformable convolution, PCD follows a coarse-to-fine manner to handle large and complex motion. In EDVR[10], the Temporal and Spatial Attention (TSA) module is used to fuse the aligned frames. Different temporal and spatial neighboring values are not equally informative due to occlusion, motion blur, and parallax problems. As mentioned in the FRVSR paper that the SRNet will implicitly learn to ignore the misalignment output of FNet. In EDVR[10], during fusion, the explicit weights are calculated temporally and spatially according to the frame similarity in the embedding space. This allows the fusion to focus on the better-aligned area and ignore the mis-

aligned area. Dataset bias is also tested in EDVR[10] paper, if the training set and test set aren't sampled from the same source, there is a 0.5~1.5dB drop in performance.

1.1.2.4 FRVSR

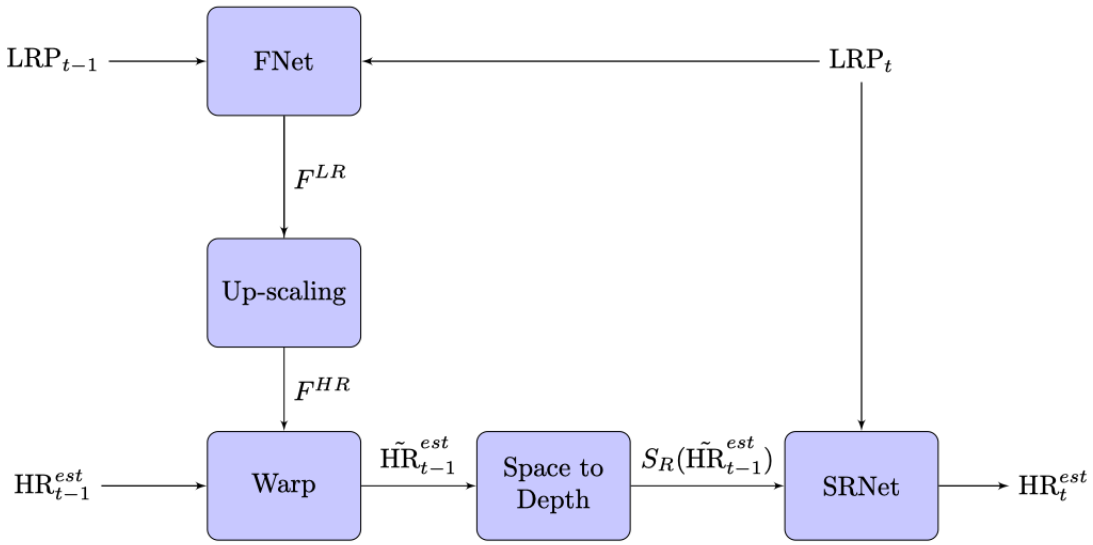

Figure 1.1: FRVSR structure

As part of our work is based on FRVSR[11], we will have a detailed explanation about how FRVSR works in this section.

As shown in Figure 1.1, FRVSR propagates the high-frequency information by feeding the HR estimation back to the next time step for alignment. It uses explicit optical flow to align the previous HR estimation frame to the current frame. The SRNet is trained to fuse the information from aligned previous estimated HR frame and current LR frame. The recurrent structure is used instead of a sliding window to propagate information in a single direction without limitation of the window size.

Figure 1.1 shows the original framework of FRVSR with pristine LR (LRP) as input. In this implement, we use the scaling factor $R = 2$ for the later integration with the SHVC codec. Therefore the output will be 2 times larger than the input. Two CNN networks, FNet and SRNet, are

used to compose the RNN structure. The first network is a Flow Net (FNet) that calculates the optical flow from the 2 consecutive LR input frames and provides motion information for the SR-Net. The structure of FNet is shown in Figure 1.2. FNet follows an encoder-decoder structure. The spatial size of the feature map first shrinks 8 times by the max-pooling layer in the encoding process. Then they are upscaled to the original size using bilinear interpolation in the decoding process. The depth of the channel number doubles when the spatial size shrinks. The output F^{LR} is an optical flow of shape $H \times W \times 2$ from the last $3 \times 3 \times 2$ convolution layers followed by Tahn activation that represents the pixel position mapping between two frames.

$$F^{LR} = \text{FNet}(\text{LRP}_{t-1}, \text{LRP}_t) \in [-1, 1]^{H \times W \times 2} \tag{1.1}$$

LRP_t is the input pristine LR frame at time t. For each pixel in the LR, the FNet try to find a corresponding pixel in LRP_{t-1}.

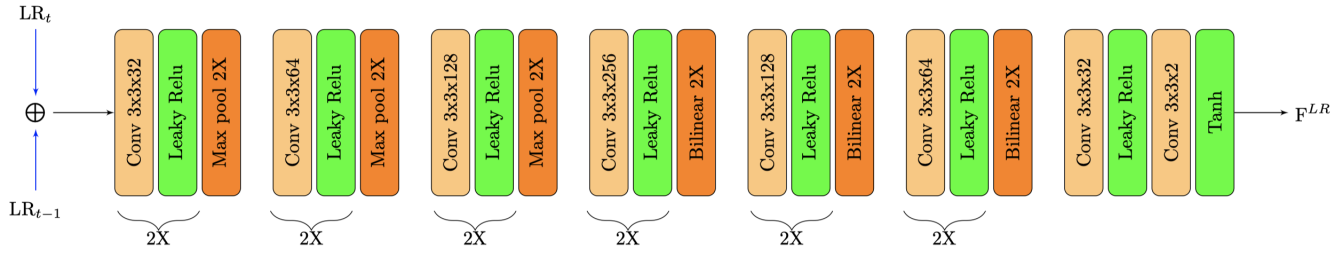

Figure 1.2: FRVSR FNet

The loss function of the FNet is the L2 distance between the previous LR frame warpped with motion and the current LR frame.

$$L_{\text{flow}} = ||\text{WP}(LR_{t-1}, F^{LR}) - LR_t||_2^2 \tag{1.2}$$

WP is the warping operation introduced in Spatial Transformer Networks[15].

The output of FNet is in LR space. The flow is upsampled R times using bilinear interpolation before being warped with the previous estimated image in the HR space.

To use the information in the previous frames without introducing too much calculation, the previous estimation of HR frame HR_{t-1}^{est} is used as feedback for the current time step. To align the previous frame with the current frame we need to move the pixels in the previous frames according to the up-scaled optical flow given by the FNet. The output is of shape $RH \times RW$.

$$\tilde{\text{HR}}_{t-1}^{est} = \text{WP}(\text{HR}_{t-1}^{est}, F^{HR}) \tag{1.3}$$

The computation in the LR space is much easier than in the HR space, many SISR methods put the upsampling at the end of their network to reduce the computation cost. In this method, the calculation is also in the LR space. Now that we have the estimated HR frame after warping (past HR frames + upsampled motion between two frames), a space to depth conversion is implemented to convert the image from HR to LR space. And then the converted image is input into the SRNet together with the current LR frame.

The second network is a super-resolution network (SRNet) that uses ResBlocks to accelerate training and transpose convolution for upscaling as shown in Figure 1.3. The space to depth image is stacked with the current LR image as the input of the super-resolution network.

$$\text{HR}_{t}^{est} = \text{SRNet}(\text{LR}_t, S_R(\tilde{\text{HR}}_{t-1}^{est})) \tag{1.4}$$

S_R is the space to depth operation with R as the scaling factor.

The loss function of SRNet is the MSE between the output and ground truth HR.

$$L_{\text{sr}} = ||HR_t^{est} - HR_t||_2^2 \tag{1.5}$$

The final loss of FRVSR is the summation of the loss of FNet and SRNet.

$$L = L_{\text{sr}} + L_{\text{flow}} \tag{1.6}$$

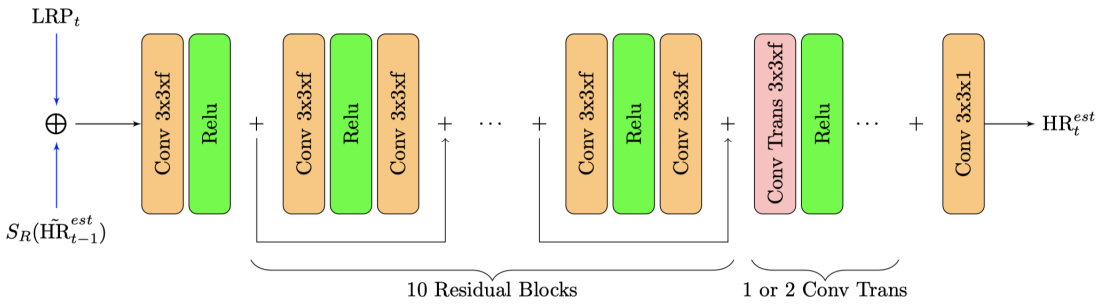

Figure 1.3: FRVSR SRNet

By adding an optical flow estimation network before the traditional SISR network and feeding the previous HR estimate and current LR input to the super-resolution network, the motion information can be carried through multiple frames. This method avoids using input LR frames repeatedly in the sliding window to save computations. Also, the output HR estimation frames will keep inter-frame temporal dependency due to the optical flow and the explicit motion warping.

However, there are some drawbacks to this model.

1. The explicit optical flow method tries to find pixel mapping between current and previous LR frames which may not always exist in case of occlusion and large motion.

2. The only way information can be propagated is through the explicit motion compensation of the previous SR frame, so whenever occlusion appears the information of the background object is lost.

1.1.2.5 RLSP

Like FRVSR, RLSP[12] also keeps the information in the SR frame and feedback to the next step. However, besides the HR$_{est}$ feedback, RLSP adds another hidden states to the recurrent sturcture. This implicit latent feature space can propagate information through multiple frames and let the network learn what needs to be saved in the latent space for future use. For example, when occlusion occurs at the current frame the features about the covered object can still be saved in the latent space and be propagated to the future frames. This is what FRVSR can't do using only

the HR_{est} as feedback.

In the RLSP model, 3 local LR frames, the previous hidden state and previous HR estimation frame are concatnated and feed into the current time step to produce the next hidden state and the current HR estimation. The explicity motion compensation in FRVSR is removed and RLSP network itself will learn whether to use previous frame as reference. On one hand, the complexity of FNet is saved. On the other hand, with the help of the hidden state, the model can better handle the cases where no exact motion matching between frames.

Although RLSP is better than FRVSR on VSR tasks, designing a deartifact module for RLSP is more difficult as the RLSP Cell is tightly integrated without explicit alignment. So we will focus on adapting FRVSR for our scalable codec.

1.2 Scalable Video Coding

SHVC[16] is the scalable extension of HEVC with different types of scalability. The hierarchical temporal prediction structure of HEVC provides the temporal scalability. SHVC supports bit depth, spatial, coarse grain SNR, interlaced-to-progressive, color gamut and external base layer scalabilities etc. We will focus on spatial scalability in this work.

The base layer (BL) of SHVC is a normal HEVC bitstream, by decoding the enhancement layer (EL) we can have higher frame rate, larger resolution, and better quality videos. There are two main benefits of using SHVC as a codec:

1. A single bitstream can fit different types of customer devices by only transferring the required EL through the network. This will reduce the complexity for the streaming provider to manage different encoding versions of the same video and reduce the network bandwidth.

2. When coding the current EL pictures, the collocated reconstructed pictures (resampled if necessary) from the BL can be used as inter-layer reference pictures (ILR). Due to this reference structure of SHVC the size of the scalable bitstream is much smaller than the sum of two separate bitstreams of different resolutions. Therefore the storage cost can be saved.

SHVC requires only high-level syntax changes. Changes are restricted to slice header level and above. Therefore, the EL codec in SHVC does not allow low-level (block-level) changes to the single-layer HEVC design.

The original input videos are downsampled and coded into the base layer (BL) bitstream using a conformant HEVC codec or a non-HEVC external (e.g., AVC) codec. To code the enhancement layer (EL) pictures, the coded BL videos in the lower reference layer are used for inter-layer prediction to improve EL coding efficiency. In SHVC, both reconstructed videos and motion parameters from the reference layer can be used for inter-layer prediction.

Besides the normal block-based hybrid video coding framework, the inter-layer reference pictures are used as references in each EL. The inter-layer processing includes 3 steps:

1. resampling process

2. color mapping process

3. motion field mapping process

The color mapping process is for color gamut scalability which we won't use in our project. We want to replace the resampling process with a machine-learning-based super-resolution method to provide a high-quality reference for EL and to save the encoding cost. In our work, X265 will be used as BL encoder to utilize its better rate control algorithm and faster encoding speed. The BL motion fields will be imported from X265 and the resampling process will be replaced by VSR model.

1.2.1 The resampling process

1.2.1.1 SHM Downsampler and sampling grid

In SHVC, videos of BL resolution and EL resolution are needed as input. The downsampling process from EL to BL is defined in [16] and the derivation of the filter coefficients is in [17]. 12-tap 2D separable downsampling filters are applied to the EL video. We called this filter "SHM" as it was provided in the reference software. Lanczos and Gaussian filters are also tested in section 3.2 to make the best trade-off between sharpness and aliasing.

There are 2 types of sampling grid positions. The default one is named "zero alignment" where top-left pixels after applying the low-pass filter are kept during decimation. Another is named "center alignment" where the middle pixels for the resampling factor are kept. Figure 1.4 is an 2X downsampling example on YUV420 format. The squares are the luma pixels and the yellow squares are the ones after downsampling. The circles are the chroma pixels, and the yellow circles are the ones after downsampling. In the YUV420 format, 4 luma pixels share 1 chroma pixel. The sampling grid has to be signaled in the bitstream to make sure the upsampling process on the decoder uses the same position. In our work, we choose the center alignment position for the downsampling process as FFMPEG default resampling operations are center-aligned.

The VSR model will adapt to the sampling grid depend on whether zero alignment or center alignment is used when generating the training set. But the downsampling process and upsampling

process have to use the same sampling grid.

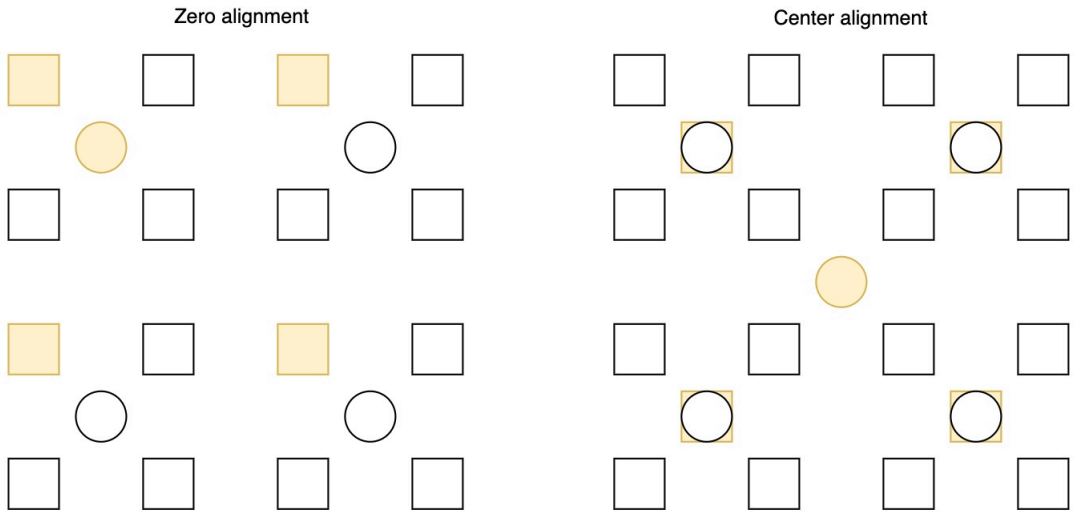

Figure 1.4: Sampling grid position on 2X downsampling example

1.2.1.2 SHM upsampler DCTIF

A 16 times upsampling filters are defined in H.8.1.4.2.2 in the HEVC standard [18]. The upsampling process in SHM uses discrete cosine transform (DCT) as the upscaler to be backward compatible with the fractional motion estimation in HEVC. At 1/2 and 1/4 pixel positions, the filter coefficients are the same as the HEVC fractional interpolation filters. By defining 16X upsampling filter, SHVC can support arbitrary scaling factors between layers. But not all 16X interpolation are calculated in the real implementation, only those needed after decimation are calculated.

2. PROPOESED METHOD

2.1 Modification on VSR for the scalable codec

SISR won't introduce any delay and buffering if integrated into SHVC. But the generated Inter-Layer Reference picture (ILR) won't be temporal consistent and the performance of SISR can be further improved by exploiting the temporal information in videos. Therefore, we are going to use the VSR model. However, there are some differences between normal video super-resolution task and the super-resolution in scalable video codec. In order to use ML-based VSR to replace the DCT upsampling in SHVC, the SR model needs to be modified for this task accordingly. The differences are explained as follows.

2.1.1 Different purpose

Sometimes HR videos might have large motion blur or out of focus area. The task of our VSR method is not to generate a photo-realistic frame, but to generate frames as close to HR as possible for the enhancement layer to reference.

GAN-based SR might increase the residual of EL because the hallucinated textures are usually far from the ground truth. The VMAF score of GAN-based VSR might be high enough so that EL encoding is not necessary. However, we can't switch between the GAN-generated video and EL encoded video as they aren't temporal consistent, and to generate GAN-based VSR result, the display order has to be followed leading to a large delay in encoding.

2.1.2 Different data format

Our input frame is a BL reconstructed image (LRC) instead of a pristine LR input (LRP). The compression artifact is added to the BL image in the process of transformation and quantization during BL encoding with different coding parameters. A de-artifact module needs to be designed before performing super-resolution to address the BL compression artifact. Our DANet is introduced in 2.2.

Constant rate factor (CRF) is a constant quality encoding mode in X265. The BL encoder will

try to generate the compressed video of the same quality without any bitrate requirement. Different CRF setting shows different levels of artifact on the training set. We use the mixed CRF setting to generate our training set.

Most of the existing SR datasets don't have LRC data. It is also not appropriate to generate the LRC from the existing datasets. For example, the 7-frame short clips in the Vimeo90k dataset doesn't have enough length to represent the compression process on a normal video as it only contains one GOP of size 6.

Another issue is that in the video codec, the common video format is YUV420. The chroma channels are downsampled 2X therefore smaller than the luma channel. However, for most of the other VSR models, the input and output are both in RGB format. In our experiment, the model's input and output are both luma channel only.

Different training sets and different data formats (Y only) make the benchmark with other VSR methods difficult. Therefore we need to generate our own training set and test set. The detailed dataset generation is in Section 3.1.

2.1.3 Different processing order

SHVC adopts the multi-loop decoding framework. Pictures in different layers of the same frame are coded into the bitstream in ascending order of layer indices which means that the encoding order is first BL then EL of the same POC. With random access profile, the GOP structure determines which POC is encoded first. However, the VSR algorithm, especially recurrent VSR like FRVSR and RLSP, usually takes frames as input by display order. Therefore, the discrepancy between the encoding order and display order leads to the delay and buffering in the scalable encoder and decoder.

For example, a GOP length of 6 from X265 is shown in Figure 2.1 and Table 2.1. The encoding order is 0-6-3-1-2-4-5. But in the normal VSR model to get the SR result of the 6th frame all previous frames are needed. However, in SHVC, when encoding the 6th frame, frames 1 to 5 haven't been encoded yet and we don't have the LRC input to generate the SR result for the 6th frame.

The recurrent VSR model like FRVSR and RLSP usually provide better VSR performance, longer temporal support, and lower computational requirement. Usually the display order is followed when generating the SR result as inter-layer reference pictures (ILR). We will compare the effect of different temporal propagation methods with this baseline method.

As a baseline experiment, when encoding and decoding a GOP, the two-pass method is used. The first pass is to generate the ILR from the reconstructed BL. Then the VSR results are generated following the display order. The second pass is to encode the EL based on VSR ILR. This integration of VSR and SHVC is at video level instead of frame level and it will increase the memory needed for buffering in a real application. This method is named "noTID" meaning no temporal layered structure is used.

We proposed three different methods. First, we can execute VSR follow the encoding order where the previous frame feedback is the previous frame in encoding order. For example, in Figure 2.1, the encoding order is 0-6-3-1-2-4-5. This method is named "encodingTID".

Second, we can treat different temporal layers as separated videos and propagate hidden states only inside the same temporal layer. For example, frames 0-6-12-18 are from temporal layer 0. Frames 3-9-15-21 are from temporal layer 1 and frames 1-2-4-5-7-8-10-11 are from temporal layer 2. This method is named "separateTID".

Third, we can use the previous frame whose TID is not larger than the current frame's TID as feedback. This method will ensure the nearest available frame is used for the current frame's SR. For example, frame 0-1-2, 0-3-4-5, 0-6-7-8 will be treated as separate videos. This method is named "nestedTID".

The experiment results are disscussed in section 3.5. The perfermance order from best to worst is no TID, nested TID, encoding TID, separate TID.

2.2 FRVSR with DANet

To integrate the VSR model into the SHVC codec, the decoder only has access to the reconstructed frames instead of the pristine frames before quantization which are normally used in other VSR models. Therefore the FRVSR structure needs to be modified as the compression process

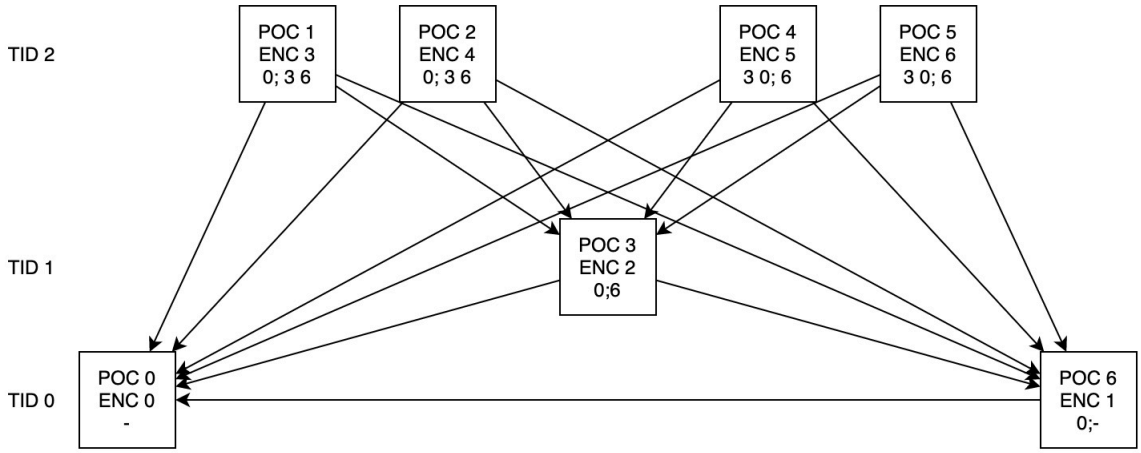

Figure 2.1: GOP structure of 6 frames in X265 BL, POC is frame display index, ENC is the encoding order

Encode Order	Type	POC	List 0	List 1
0	I-SLICE	0	-	-
1	P-SLICE	6	0	-
2	B-SLICE	3	0	6
3	b-SLICE	1	0	3 6
4	b-SLICE	2	0	3 6
5	b-SLICE	4	3 0	6
6	b-SLICE	5	3 0	6
7	P-SLICE	12	6 3 0	-
8	B-SLICE	9	6 3 0	12
9	b-SLICE	7	6 3	9 12
10	b-SLICE	8	6 3	9 12
11	b-SLICE	10	9 6 3	12
12	b-SLICE	11	9 6 3	12
13	P-SLICE	18	12 9 6	-

Table 2.1: An X265 reference structure example of one GOP of 6 frames

added artifacts to the input. We propose the DANet structure as shown in Figure 2.2.

FNet is the same as the original FRVSR but the input is the compressed LR (LRC) in the

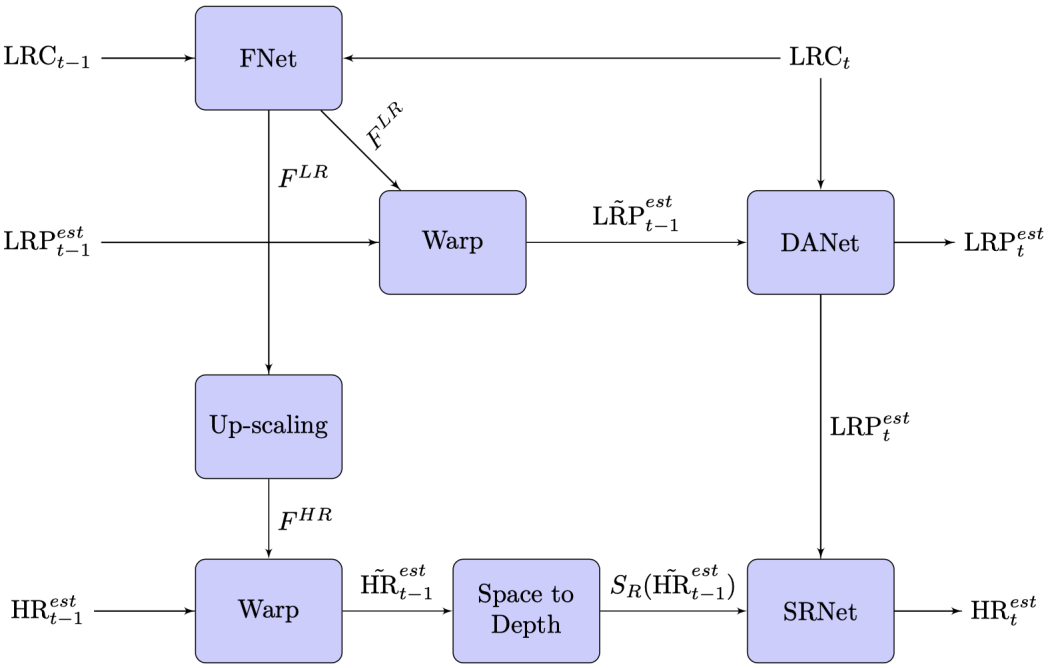

Figure 2.2: FRVSR with DANet

modified FRVSR with DANet. The compression is done by X265.

$$F^{LR} = \mathrm{FNet}(\mathrm{LRC}_{t-1}, \mathrm{LRC}_t) \in [-1, 1]^{H \times W \times 2} \tag{2.1}$$

The de-artifact module DANet is added before the SRNet and trained using LRP as ground truth. The DANet also follows the recurrent structure by using the feedback of LRP estimation from the previous time step.

By directly using the optical flow from FNet, warping is also performed in the LR space on the previous output of the DANet which is the LRP estimation LRP^{est}_{t-1} to generate the LRP estimation of the current frame LRP^{est}_t. The warping operation is expressed in Equation 2.2.

$$\tilde{\mathrm{LRP}}^{est}_{t-1} = \mathrm{WP}(\mathrm{LRP}^{est}_{t-1}, F^{\mathrm{LR}}) \tag{2.2}$$

Then the warped LRP estimation is concatenated with the current input LRC_t as the input of

DANet. The output of DANet is the estimated pristine version of the LRC image. Then it is feed into SRNet together with the aligned previous HR estimation as in FRVSR to generate the final SR result.

$$\text{LRP}_t^{est} = \text{DANet}(\text{LRC}_t, \tilde{\text{LRP}}_{t-1}^{est}) \tag{2.3}$$

A loss term is also added to the total loss for the DANet.

$$L_{da} = ||\text{LRP}_t^{est} - \text{LRP}_t||_2^2 \tag{2.4}$$

The total training loss of the modified FRVSR is

$$L = L_{sr} + L_{flow} + L_{da} \tag{2.5}$$

The network structure of SRNet, FNet, DANet are shown in Figure 1.3, Figure 1.2, Figure 2.3. They are all fully convolutional networks and do the calculation only in LR space. All the kernels are of size 3 and stride 1. The number of feature channels f is 128 in our experiment.

In the SRNet, the transposed convolution layers are of stride 2 to upscale the input feature map. 1 or 2 transposed convolution layer can be used depends on whether the scaling factor is 2 or 4.

In the FNet, the encoder and decoder structure is used. The activation function in FNet is LeakyReLU of leakage factor of 0.2 and 2X represents the same layers appear twice.

In both DANet and SRNet, the ResBlocks are used to accelerate the training. The DANet only operates in LR space therefore no transposed convolution layers are used. 3 ResBlocks are used in DANet for faster computation. Only the Y channel is output by the last output layer.

2.3 Codec Integration

X265 is one of the best open-source HEVC encoders in the market. Its encoding speed is much faster than SHVC reference software by utilizing multiple CPU cores in both the wavefront parallel processing mode and the frame-level parallelism. It also uses more sophisticated rate control algo-

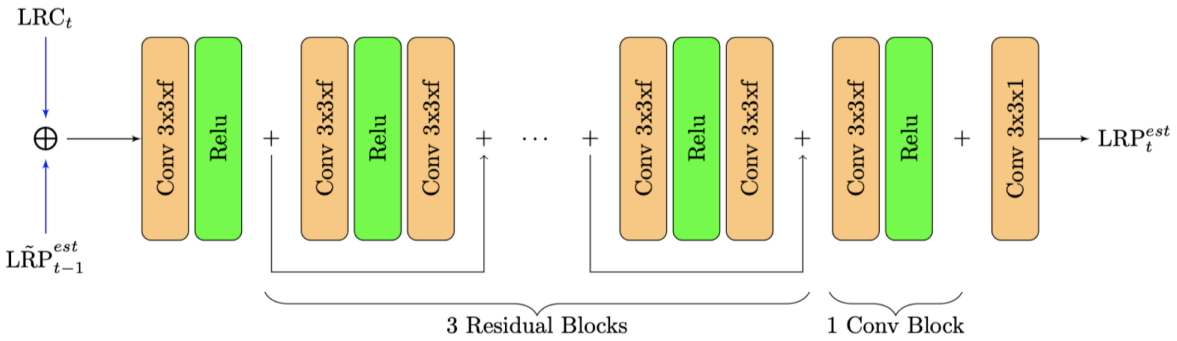

Figure 2.3: DANet structure

rithms to achieve CU-level adaptive QP. Using X265 as the base layer and builds the enhancement layer coding on top of it will give us better base layer encoding as references. Therefore, we can achieve better coding quality with a faster speed than the original SHVC reference encoder.

When encoding an EL frame in SHM, it will reference the DCT upscaled reconstructed image from BL. Therefore, the reconstructed videos (LRC) also need to be imported from X265 into SHVC.

In SHVC, TMVP will use the reference pictures' motion vectors to derive motion vector predictors. When an EL CU reference the upscaled BL image, the motion vectors of that image also need to be upscaled accordingly. This process is called Motion Field Mapping. When upsampling the MV, only the CUs encoded in inter prediction mode have MVs. Therefore the motion vectors and the associated CU encoding mode also need to be imported from X265.

To make use of the adaptive QP calculated in BL encoding, we also import the QP map from X265 and enforce the EL QP decision to be the same as the upscaled BL QP map.

Besides importing the reconstructed images, motion fields, CU coding modes and QP maps, the GOP structure of SHM and X265 also need to be the same. In SHVC reference software the GOP structure is fixed in the configuration file. While in X265, the GOP structure is usually dynamic based on when the scene cut appears in the timeline. In our experiment, we fixed the GOP size of X265 to 6 and modify the SHM codes to align the BL and EL GOP structure so that the same frame in BL and EL will have the same reference list. As a result, the reference index in the motion

vectors from X265 will refer to the same frames after imported to the SHVC codec.

We have two testing setups. The baseline follows the original SHVC structure using DCT as upscaler but X265 as BL, as shown in Figure 2.4. Another using VSR to replace the DCT upscaler as shown in Figure 2.5. The results are shown in Table 3.11 and Table 3.12.

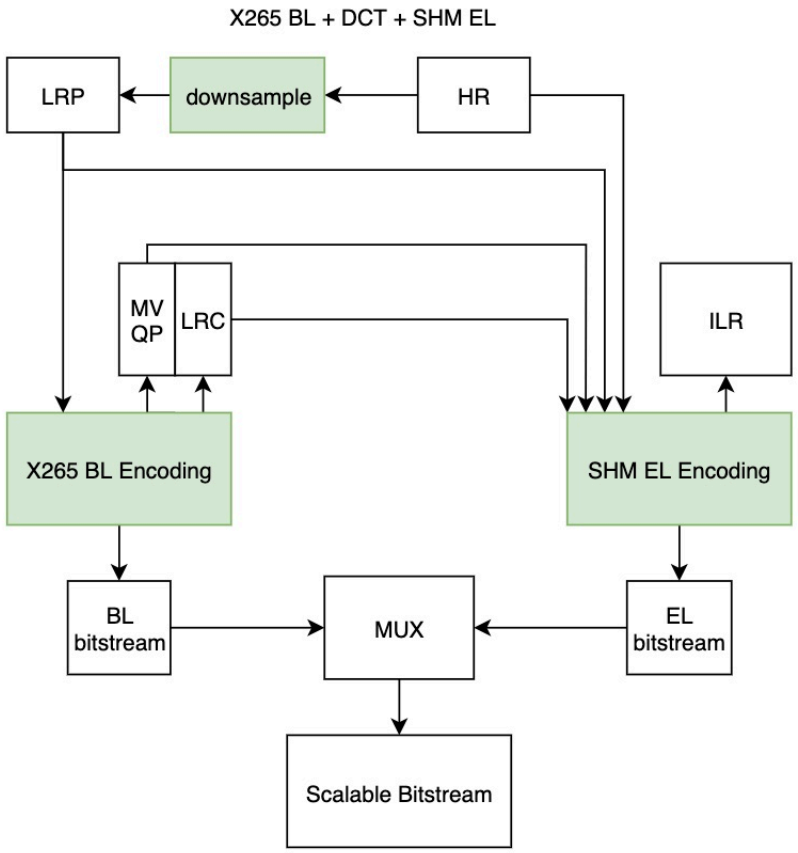

Figure 2.4: Codec integration baseline structure: X265 BL + DCT + SHM EL

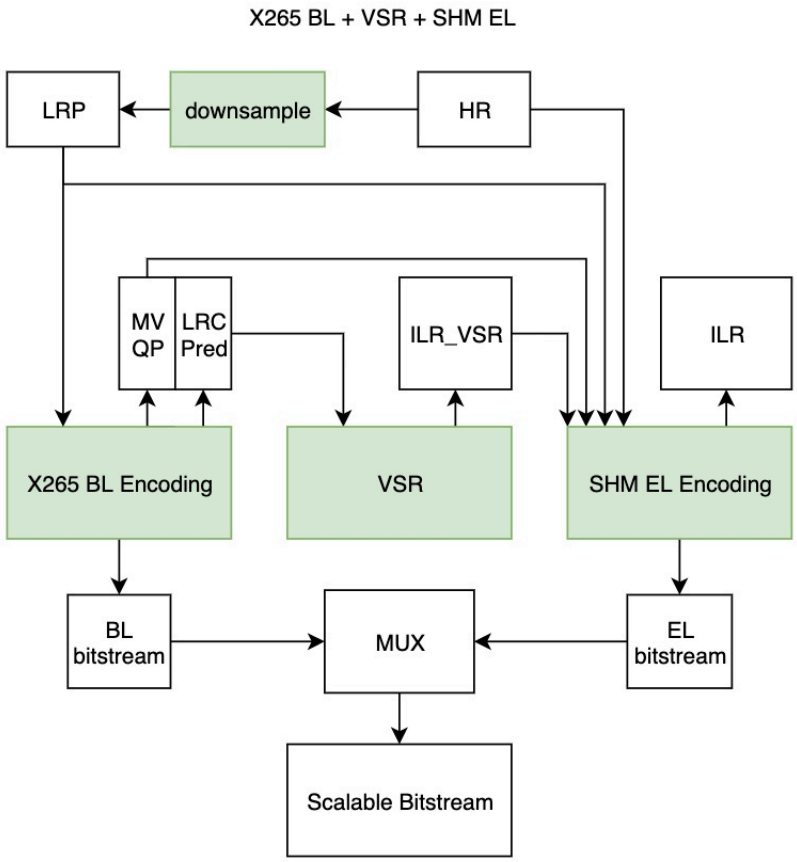

Figure 2.5: Codec integration VSR structure: X265 BL + VSR + SHM EL

3. EXPERIMENT

3.1 Data processing pipeline and experiment condition

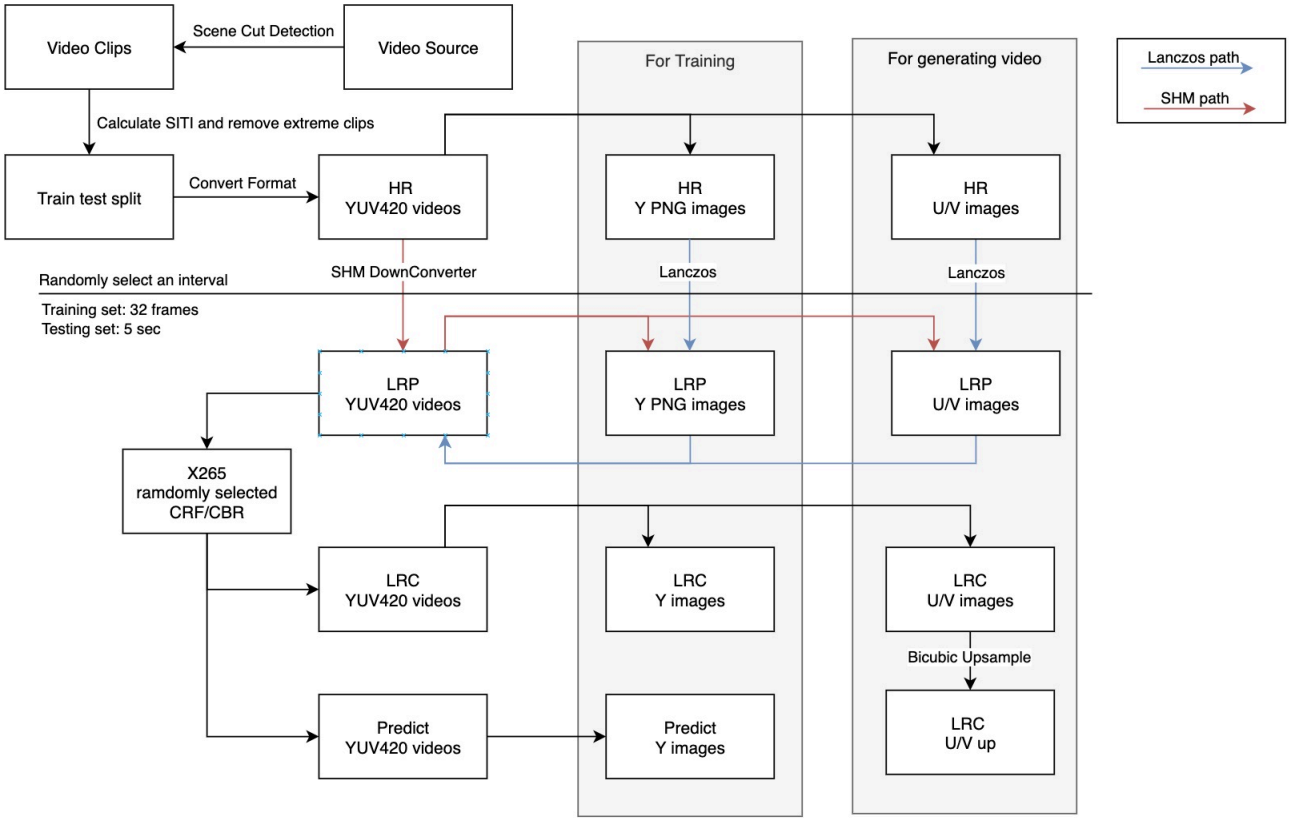

Figure 3.1: Video pre-processing pipeline

Due to different requirements in data formats as mentioned in 2.1.2. We need to generate our datasets. Our dataset is composed of few different sources that are only used for study purposes.

1. 229 one-minute video clips from Amazon Prime Video including movies, news, reality shows, sports, cartoons, and anime.

2. 7 test videos from UVG dataset [19]

3. Some Vimeo videos that are still available from Vimeo-90k dataset [20].

4. Some BBC documentaries.

Some pre-processing steps are needed to generate the training set, validation set, and test set. The whole video pre-processing pipeline is shown in Figure 3.1.

1. The scene cut detection is first performed using Davinci Resolve and then manual inspection to ensure there is no scene cut in the dataset.

2. The Spatial Information (SI) and Temporal Information (TI) are calculated for each clip to observe the distribution of the dataset on the SI-TI plane according to ITU-T P910 [21]. Some bad clips with extreme SI or TI will be removed. A test set is randomly sampled to cover the whole SI-TI plane.

3. Many clips from the same video may have repeated scenes. Therefore a subset of clips are selected from all clips to ensure heterogeneity of the training set.

4. The video clips are then converted to YUV420 format with limited range and saved as YUV raw video files.

5. To generate the LRP data, we use FFMPEG to do center-aligned Lanczos downsampling with $a = 2$. There are different downsampling kernels available such as SHM, Gaussian, and Lanczos. In the downsampling step, the videos are convolved with the kernel before decimation which will reduce the aliasing. The comparison of different downsampling kernel is in Section 3.2. We choose Lanczos with $a = 2$ as our downsampler for the codec integration.

6. To generate the LRC data we use X265 to compress the LRP raw videos. We take the constant quality encoding of X265 with CRF randomly chosen from a specific range (22 to 28). GOP length of 6 is enforced by setting "bframes" to 5. The temporal layer is also enabled. Therefore the training set compression setting is aligned with the SHM EL coding setting.

Upsampler	lrType	downFilter	metrics	calendar	city	foliage	walk	MEAN
Bicubic 2X	LRP	gaus_std0p8_center	PSNR	21.17	25.97	24.43	27.19	24.69
Bicubic 2X	LRP	shm_center	PSNR	23.77	29.21	28.51	32.26	28.44
Bicubic 2X	LRP	lanc_a2_center	PSNR	23.86	29.58	28.63	32.35	28.60
Bicubic 2X	LRP	gaus_std0p8_center	VMAF	30.18	35.63	44.57	46.24	39.15
Bicubic 2X	LRP	shm_center	VMAF	81.36	81.97	94.96	96.60	88.72
Bicubic 2X	LRP	lanc_a2_center	VMAF	75.82	77.76	89.64	91.41	83.66

Table 3.1: Comparision of different downsampling filter on 2X LRP on VID4 test set

7. The LR input needs to be cut to fit the requirement of the network. The FNet goes through an Encoder-Decoder tunnel. The spacial size is reduced 8 times by the Max Pooling layer and then recovered to its original size by the Bilinear Layer. So the LR input size has to be divisible by 8. Therefore the size of HR images has to be divisible by $R \times 8$. In the video image loader at training time, the training clips will be cut from a random position with size of 64×64 by 10 frames as LR input.

3.2 Different downsampler

To generate LRP from HR, an appropriate downsampling method is needed. The choice between different downsamplers is a trade-off between sharpness and aliasing. In SHM we want the BL to be sharper and with less aliasing. For anti-aliasing purposes, the larger the scale factor the smaller the passband of the filter should be.

A Gaussian filter has less aliasing but is too smooth for the codec because the BL is still needed to be viewed separately from EL in a scalable codec.

The Lanczos filter can produce sharper images than gaussian but will introduce some aliasing. The downsampled image will be used as the learning target for DANet. If too many aliasings are introduced then it will also affect the final SR result.

The SHM downsampling filter is defined in [17]. Its sharpness and aliasing are between gaussian and Lanczos.

In Table 3.1, we compared the 3 downsamplers using the same bicubic upsampling in FFMPEG on VID4 test set. The SHM downsampler is of the highest VMAF while the Lanczos (a=2) is of

PSNR	test_lanc	test_shm
train_lanc	39.45	39.23
train_shm	39.15	39.59

Table 3.2: DANet PSNR performance on PRIME7 CRF23 LRC of training and testing on Lanczos a=2 and SHM downsampler

VMAF	test_lanc	test_shm
train_lanc	93.78	96.27
train_shm	90.40	94.23

Table 3.3: DANet VMAF performance on PRIME7 CRF23 LRC of training and testing on Lanczos a=2 and SHM downsampler

the highest PSNR. Gaussian with 0.8 standard deviation is the lowest among the three.

We trained our model on Lanczos downsampled and SHM downsampled LRC and then test it on SHM and Lanczos respectively. The results are shown in Table 3.2 and 3.3. For PSNR, the model has the best performance when the training downsampling method and testing downsampling method are the same. For VMAF, the highest score is when the model is trained on Lanczos and tested on SHM downsampler. Therefore we take this option to generate our codec integration result.

3.3 SR results on 2X SHM downsampled LRC with CRF23

We benchmarked different 2X upscaling methods including the DANet on 7 clips test set from prime video on PSNR and VMAF[22] in Table 3.4. The inputs of this test are SHM-2X-downsampled and then compressed using X265 at CRF 23 with a fixed GOP structure of 6 frames.

1. Bicubic downsampling with a=-0.75.

2. DCTIF, discrete cosine transformation interpolation filter which is used in the SHVC as inter-layer upscaling method.

3. RAISR[7], the single image super-resolution method brought up by Google, retrained on our

modelName	metrics	41	59	71	87	106	175	185	MEAN
Bicubic	PSNR	35.42	41.50	39.31	42.26	39.37	32.63	34.53	37.86
DCTIF	PSNR	36.06	41.52	39.47	42.29	39.70	33.31	35.02	38.19
RAISR	PSNR	36.43	41.22	39.18	41.81	39.55	33.75	35.27	38.17
FRVSR	PSNR	37.89	41.20	39.48	42.30	40.18	35.90	35.67	38.95
DANet	PSNR	37.80	41.60	39.72	42.65	40.43	36.59	35.79	39.23
Bicubic	VMAF	87.49	91.31	92.43	90.25	98.88	85.62	92.27	91.18
DCTIF	VMAF	89.28	91.69	93.09	90.72	98.92	88.10	93.80	92.23
RAISR	VMAF	90.74	90.97	92.36	90.16	98.92	91.05	94.20	92.63
FRVSR	VMAF	95.02	92.73	94.87	92.90	99.04	97.22	96.47	95.46
DANet	VMAF	96.13	93.25	96.18	94.44	99.10	97.44	97.37	96.27

Table 3.4: Comparision of different upsampling method on 2X SHM downsampled CRF23 LRC on PRIME7 test set

dataset using LRC as input. No de-artifact modification.

4. FRVSR, the original FRVSR model which trained on LRP input.

5. DANet, the modified version of FRVSR with the de-artifact module.

As shown in Table 3.4, on the test set PRIME7, the DANet achieved 1.37db PSNR and 5.09 VMAF gain over bicubic. When compared with the original FRVSR where no de-artifact model is used, the difference is 0.28db PSNR and 0.18 VMAF which shows the DANet training with LRP video are better than the original VSR model.

Figure 3.2: SR result on calendar in VID4, from left to right are: Bicubic, FRVSR, DANet, HR

The videos in the PRIME7 test set are higher quality 1080p videos and 2X is not a large scaling factor therefore even the bicubic is of 37.86db PSNR and 91.18 VMAF. As shown in Table 3.5, on

modelName	metrics	calendar	city	foliage	walk	MEAN
Bicubic 2X	PSNR	23.51	28.55	27.75	30.98	27.70
FRVSR	PSNR	25.16	31.27	29.01	32.21	29.41
DANet	PSNR	25.23	31.17	29.08	32.57	29.51
Bicubic 2X	VMAF	76.34	74.72	86.95	87.16	81.29
FRVSR	VMAF	87.43	88.60	93.54	93.56	90.78
DANet	VMAF	88.53	89.06	94.23	94.56	91.60

Table 3.5: Comparision of different upsampling method on 2X SHM downsampled CRF23 LRC on VID4 test set

model	LR Type	parameters	metrics	41	59	71	87	106	175	185	MEAN
FRVSR	LRP	4851939	PSNR	40.00	46.51	43.21	47.78	42.73	37.21	39.92	42.48
FRVSR	CRF23	4851939	PSNR	37.89	41.20	39.48	42.30	40.18	35.90	35.67	38.95
DANet	CRF23	5889892	PSNR	37.80	41.60	39.72	42.65	40.43	36.59	35.79	39.23
FRVSR	LRP	4851939	VMAF	99.89	99.99	99.93	99.83	99.95	99.51	99.99	99.87
FRVSR	CRF23	4851939	VMAF	95.02	92.73	94.87	92.90	99.04	97.22	96.47	95.46
DANet	CRF23	5889892	VMAF	96.13	93.25	96.18	94.44	99.10	97.44	97.37	96.27

Table 3.6: Comparison of FRVSR on LRP/LRC and DANet on LRC

a lower resolution test set VID4, the DANet improvement from bicubic is 1.81db PSNR and 10.31 VMAF while the advantage given by the de-artifact structure gives 0.10db PSNR and 0.82 VMAF. When the video quality is lower there are more rooms for the VSR model to improve. An example comparison is shown in Figure 3.2. DANet result has less artifact than FRVSR.

3.4 The effect of compression and the gain of DANet

After using LRC as input the original FRVSR's performance drops 3.53db and 4.41 VMAF compared to the model's performance on LRP. However, by introducing the DANet, the impact of the BL compression is reduced to 3.25db and 3.6 VMAF at the expense of additional 1 million parameters, as shown in Table 3.6.

We also show the result of LRPest vs. HRest on DANet when tested with SHM downsampler and CRF23 LRC in Table 3.7. The LRPest can achieve 42.69db and 98.25 VMAF which shows that the DANet can output high-quality LRP estimation for the SRNet to do super-resolution. The final HRest is a serial combination of de-artifact and super-resolution result.

type	metrics	41	59	71	87	106	175	185	MEAN
HRest	PSNR	37.80	41.60	39.72	42.65	40.43	36.59	35.79	39.23
LRPest	PSNR	43.44	42.89	42.89	43.67	43.76	42.97	39.19	42.69
HRest	VMAF	96.13	93.25	96.18	94.44	99.10	97.44	97.37	96.27
LRPest	VMAF	97.13	98.54	98.43	97.25	99.76	97.80	98.84	98.25

Table 3.7: Comparison of DANet's LRPest and HRest

Processing Order	Metric	calendar	city	foliage	walk	MEAN
noTID	PSNR	25.46	31.62	29.43	32.88	29.85
nestedTID	PSNR	25.40	31.45	29.37	32.84	29.76
encodingTID	PSNR	25.40	31.39	29.33	32.82	29.73
separateTID	PSNR	25.37	31.31	29.32	32.80	29.70
noTID	VMAF	84.59	88.77	91.35	91.61	89.08
nestedTID	VMAF	84.34	88.32	91.10	91.46	88.81
encodingTID	VMAF	84.35	88.31	90.98	91.37	88.75
separateTID	VMAF	84.17	87.98	90.91	91.34	88.60

Table 3.8: Comparison of DANet with different processing order on Lanczos-downsampled CRF23 VID4 test set

3.5 Different processing order

As shown in Table 3.8, the display order (noTID) has the best performance as expected. The nestedTID is the 2nd because it has less reordering than encodingTID and still utilize the nearest available references to generate the current SR frame. Although separateTID doesn't have reordering, the temporal distances between frames can be as large as 1 GOP, therefore it has the worst performance. In a real application, VSR follows nestedTID order can reduce the BL input buffering with a small VSR performance drop of 0.09dB and 0.27 VMAF.

3.6 SHVC integration

After having the VSR result with the de-artifact feature we can use it to replace the DCTIF in SHM. There are 7 different combinations depends on whether the VSR or DCT is used as the upscaler and whether the EL encoding is applied.

1. BL+DCT: This is the inter-layer reference picture (ILR) inside of SHVC. The bitrate is the X265 BL bitrate and the distortion measures the upsampled video by DCTIF.

2. BL+FRVSR: This is the ILR that we used to replace DCTIF. The VSR reference video is generated by original FRVSR trained on LRP as a VSR baseline. The bitrate is the X265 BL bitrate and the distortion measures the upsampled video by FRVSR.

3. BL+DANet: Same as BL+FRVSR, but the VSR reference video is generated by DANet to test the modified DA module. The bitrate is the X265 BL bitrate and the distortion measures the upsampled video by DANet.

4. BL+DCT+EL: The SHVC uses the combination of base layer coding, DCTIF upsampling, and enhancement layer coding. The bitrate is the X265 BL bitrate plus the SHM EL bitrate. The distortion measures the EL reconstructed video from SHM.

5. BL+FRVSR+EL: The modified scalable codec uses the combination of base layer coding, VSR upsampling, and enhancement layer coding to add finer details where VSR model doesn't work well. The bitrate is the X265 BL bitrate plus the SHM EL bitrate. This EL bitstream is different from the one referencing the DCT upsampled video. The distortion measures the EL reconstructed video referencing the VSR video. FRVSR is used as a VSR integration baseline.

6. BL+DANet+EL: Same as BL+FRVSR+EL, but DANet is used as the inter-layer VSR method.

7. SL: The single layer encoding. Using X265 to encode the video with BL CRF at the EL resolution.

The testing process is composed of BL X265 encoding, EL DCT SHM encoding, EL VSR SHM encoding, and SL encoding. The encoding is executed at the video level which means the whole BL video is used to generate VSR results. And the VSR result is input into the modified SHM to encode the EL.

We measure the distortion of our video in PSNR and VMAF[22]. There are 7 videos, the whole form is in Table 3.11 and 3.12. Take one video named 00041_00001243 for example, the PSNR and VMAF result is shown in Table 3.9 and 3.10 respectively. The rate-distortion (R-D) curves are plotted in Figure 3.3 and 3.4. We collect 7 data points on the R-D curve by varying the BL CRF setting to be 21,23,25,27,29,31 and 33. The BL QP map is determined by the rate control algorithm in X265. The EL QP map in SHM is 3 larger than the BL QP for each CU accordingly.

On both PSNR and VMAF our VSR method exceed the DCTIF in SHVC. The gap between ILR can be converted to the final EL bitrates saving. The final BD-rate improvement 6.14% on PSNR, 15.58% on VMAF for video 41.

In Figure 3.4, the BL+VSR VMAF metric is at the top left of the BL+DCT+EL, BL+VSR+EL, and SL R-D curve which means that it is unnecessary to add the EL encoding on top of the VSR result in the VMAF aspect. The actual bits spending on the EL coding won't give improvement on VMAF and sometimes even makes it worse. In Table 3.10, the BL+VSR+EL's VMAF only has a small gain over BL+VSR at a higher CRF setting. Only when video quality is worse due to the compression of BL, the EL encoding shows some benefit in VMAF score. But on PSNR, the EL coding is always useful as the high-frequency component can't be restored by the VSR can be added by the EL.

The SL curve is usually better than the scalable solution due to the cost of the extra semantics needed in the layered structure of the scalable codec. Even after VSR is used in the scalable codec, it is still not as good as the SL PSNR performance in most cases. But in VMAF, the VSR method is closer to the SL performance.

On PSNR and VMAF, DANet performs better than the original FRVSR on both ILR comparison and the comparison after the EL coding. Table 3.13 and 3.14 shows the PSNR and VMAF BD-Rate. In Table 3.13, FRVSR has positive BD-Rate 2.03% and negative BD-PSNR -0.06 dB which means without DANet using VSR to replace DCTIF has negative effect on scalable coding.

The content of the video also affects the result. For example, in Figure 3.5 and 3.6 video 71 shows that at very low bitrate the upscaled BL single layer is better than coding at EL resolution.

However at higher bitrates only coding at EL resolution can achieve better quality.

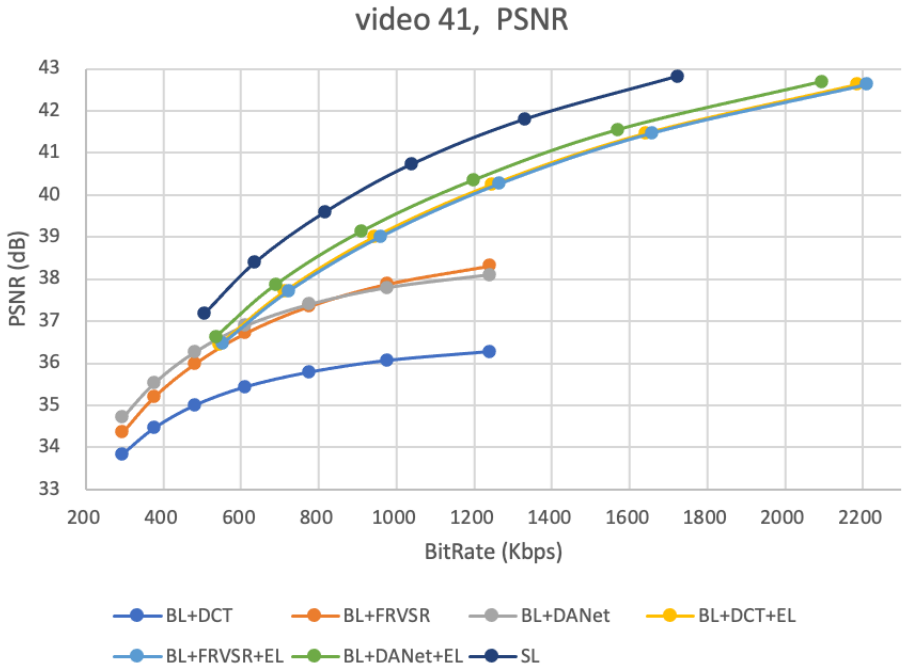

Figure 3.3: SHVC integration rate distortion graph with PSNR as quality metric on video 41

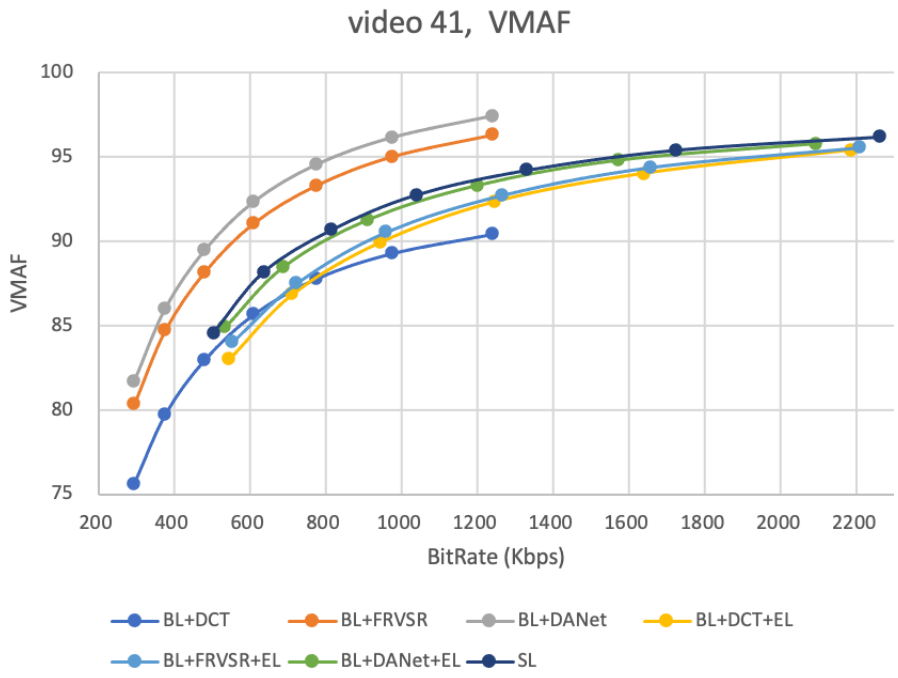

Figure 3.4: SHVC integration rate distortion graph with VMAF as quality metric on video 41

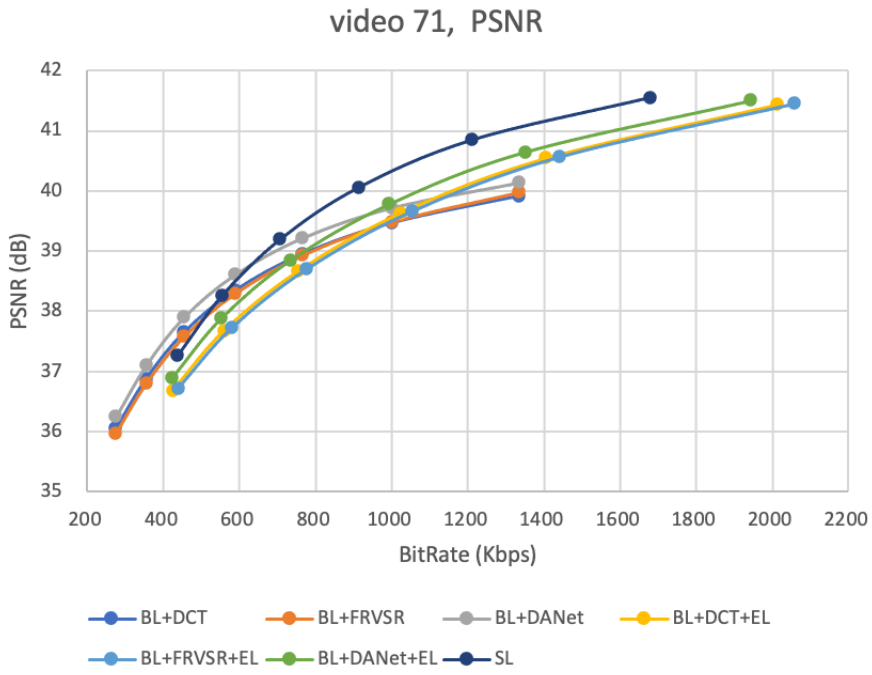

Figure 3.5: SHVC integration rate distortion graph with PSNR as quality metric on video 71

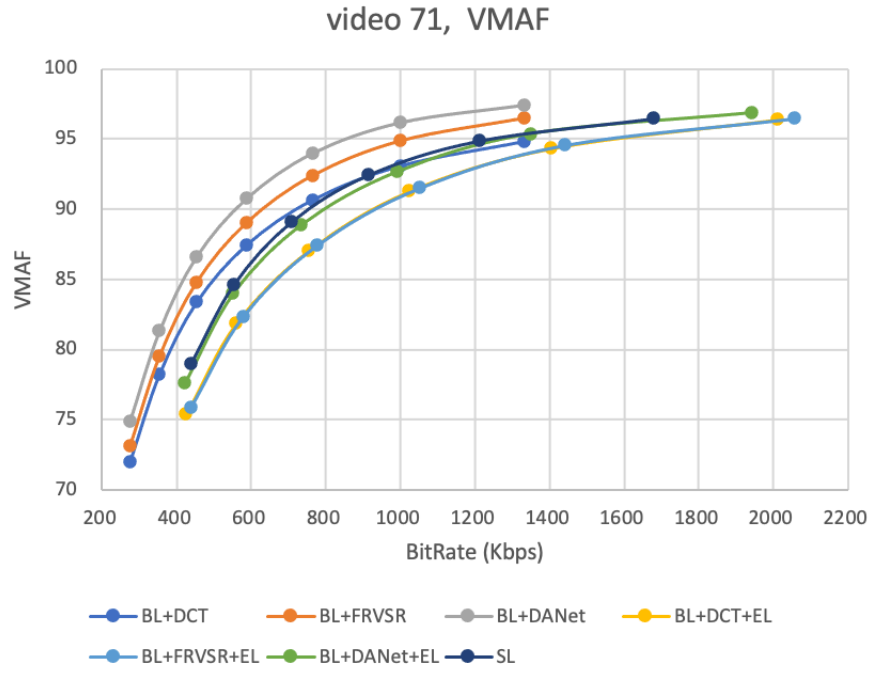

Figure 3.6: SHVC integration rate distortion graph with VMAF as quality metric on video 71

PSNR	VideoName	41						
Combination	BL EL CRF	21_24	23_26	25_28	27_30	29_32	31_34	33_36
BL+DCT	BL_bitrate	1240.47	974.83	775.18	610.46	480.87	377.13	295.40
	ILR_DCT_PSNR	36.27	36.06	35.78	35.43	34.99	34.46	33.84
BL+FRVSR	BL_bitrate	1240.47	974.83	775.18	610.46	480.87	377.13	295.40
	ILR_VSR_PSNR	38.32	37.89	37.36	36.72	35.99	35.21	34.37
BL+DANet	BL_bitrate	1240.47	974.83	775.18	610.46	480.87	377.13	295.40
	ILR_VSR_PSNR	38.11	37.80	37.40	36.89	36.27	35.54	34.72
BL+DCT+EL	DCT_totalBitrate	2977.10	2185.20	1640.43	1246.95	943.16	711.88	545.20
	recon_DCT_PSNR	43.72	42.62	41.46	40.25	39.00	37.71	36.46
BL+FRVSR+EL	VSR_totalBitrate	3005.82	2210.09	1657.98	1264.87	958.81	722.39	553.58
	recon_VSR_PSNR	43.73	42.63	41.47	40.27	39.01	37.71	36.47
BL+DANet+EL	VSR_totalBitrate	2872.28	2094.52	1571.07	1199.69	911.22	689.16	534.74
	recon_VSR_PSNR	43.79	42.70	41.56	40.36	39.14	37.87	36.64
SL	SL_totalBitrate	2262.15	1723.09	1331.35	1038.94	815.39	636.38	504.90
	recon_SL_PSNR	43.74	42.82	41.80	40.73	39.59	38.40	37.17

Table 3.9: Different SHVC integration combination on test video 41, using PSNR as the metric

VMAF	VideoName	41						
Combination	BL EL CRF	21_24	23_26	25_28	27_30	29_32	31_34	33_36
BL+DCT	BL_bitrate	1240.47	974.83	775.18	610.46	480.87	377.13	295.40
	ILR_DCT_VMAF	90.43	89.28	87.78	85.66	82.96	79.73	75.62
BL+FRVSR	BL_bitrate	1240.47	974.83	775.18	610.46	480.87	377.13	295.40
	ILR_VSR_VMAF	96.31	95.02	93.31	91.07	88.17	84.72	80.34
BL+DANet	BL_bitrate	1240.47	974.83	775.18	610.46	480.87	377.13	295.40
	ILR_VSR_VMAF	97.41	96.13	94.53	92.33	89.46	86.03	81.71
BL+DCT+EL	DCT_totalBitrate	2977.10	2185.20	1640.43	1246.95	943.16	711.88	545.20
	recon_DCT_VMAF	96.29	95.38	94.02	92.34	89.93	86.86	83.02
BL+FRVSR+EL	VSR_totalBitrate	3005.82	2210.09	1657.98	1264.87	958.81	722.39	553.58
	recon_VSR_VMAF	96.42	95.55	94.38	92.72	90.54	87.50	84.02
BL+DANet+EL	VSR_totalBitrate	2872.28	2094.52	1571.07	1199.69	911.22	689.16	534.74
	recon_VSR_VMAF	96.55	95.79	94.80	93.31	91.27	88.48	84.90
SL	SL_totalBitrate	2262.15	1723.09	1331.35	1038.94	815.39	636.38	504.90
	recon_SL_VMAF	96.20	95.41	94.22	92.75	90.66	88.15	84.57

Table 3.10: Different SHVC integration combination on test video 41, using VMAF as the metric

PSNR	Combination	BL+DCT		BL+FRVSR		BL+DANet		BL+DCT+EL		BL+FRVSR+EL		BL+DANet+EL		SL	
VideoName	BL EL CRF	bitrate	PSNR	bitrate	PSNR	bitrate	PSNR	bitrate	PSNR	bitrate	PSNR	bitrate	PSNR	bitrate	PSNR
41	21_24	1240.47	36.27	1240.47	38.32	1240.47	38.11	2977.10	43.72	3005.82	43.73	2872.28	43.79	2262.15	43.74
	23_26	974.83	36.06	974.83	37.89	974.83	37.80	2185.20	42.62	2210.09	42.63	2094.52	42.70	1723.09	42.82
	25_28	775.18	35.78	775.18	37.36	775.18	37.40	1640.43	41.46	1657.98	41.47	1571.07	41.56	1331.35	41.80
	27_30	610.46	35.43	610.46	36.72	610.46	36.89	1246.95	40.25	1264.87	40.27	1199.69	40.36	1038.94	40.73
	29_32	480.87	34.99	480.87	35.99	480.87	36.27	943.16	39.00	958.81	39.01	911.22	39.14	815.39	39.59
	31_34	377.13	34.46	377.13	35.21	377.13	35.54	711.88	37.71	722.39	37.71	689.16	37.87	636.38	38.40
	33_36	295.40	33.84	295.40	34.37	295.40	34.72	545.20	36.46	553.58	36.47	534.74	36.64	504.90	37.17
59	21_24	2373.41	42.27	2373.41	41.95	2373.41	42.27	5661.54	44.60	5942.86	44.61	5564.20	44.76	4273.37	43.91
	23_26	1792.36	41.52	1792.36	41.20	1792.36	41.60	3970.26	43.66	4157.05	43.66	3894.01	43.83	3180.58	43.10
	25_28	1355.19	40.70	1355.19	40.41	1355.19	40.83	2810.95	42.68	2943.02	42.67	2761.03	42.85	2386.40	42.24
	27_30	1022.76	39.83	1022.76	39.55	1022.76	39.98	2053.17	41.66	2139.73	41.64	2017.09	41.82	1799.48	41.33
	29_32	777.77	38.91	777.77	38.66	777.77	39.07	1495.64	40.57	1554.91	40.55	1477.88	40.74	1358.75	40.36
	31_34	588.72	37.92	588.72	37.70	588.72	38.08	1088.72	39.44	1129.29	39.42	1080.23	39.60	1033.65	39.34
	33_36	448.37	36.89	448.37	36.70	448.37	37.04	822.49	38.30	849.30	38.28	818.69	38.45	788.29	38.26
71	21_24	1333.80	39.91	1333.80	39.98	1333.80	40.14	3132.16	42.29	3187.88	42.29	3048.88	42.34	2517.59	42.16
	23_26	1001.83	39.47	1001.83	39.48	1001.83	39.72	2013.13	41.43	2057.58	41.45	1944.32	41.50	1678.98	41.54
	25_28	764.51	38.95	764.51	38.92	764.51	39.22	1405.70	40.54	1441.72	40.57	1350.91	40.64	1211.90	40.84
	27_30	587.36	38.34	587.36	38.29	587.36	38.60	1023.82	39.63	1053.30	39.66	992.87	39.78	913.86	40.05
	29_32	453.80	37.65	453.80	37.57	453.80	37.90	754.24	38.66	777.69	38.70	735.40	38.85	707.44	39.19
	31_34	354.63	36.88	354.63	36.80	354.63	37.10	559.50	37.67	579.05	37.72	552.03	37.88	553.78	38.26
	33_36	275.45	36.05	275.45	35.97	275.45	36.25	425.46	36.68	439.23	36.72	422.51	36.88	438.15	37.27
87	21_24	1461.28	42.93	1461.28	42.97	1461.28	43.28	3425.77	45.58	3504.05	45.61	3413.47	45.72	2469.62	45.41
	23_26	1097.68	42.29	1097.68	42.30	1097.68	42.65	2378.72	44.54	2431.81	44.56	2364.33	44.66	1837.43	44.55
	25_28	837.98	41.57	837.98	41.55	837.98	41.93	1699.28	43.49	1737.24	43.51	1676.11	43.60	1384.72	43.63
	27_30	638.42	40.80	638.42	40.76	638.42	41.15	1271.51	42.46	1292.01	42.45	1256.50	42.55	1054.00	42.68
	29_32	489.18	39.96	489.18	39.90	489.18	40.28	944.23	41.38	954.39	41.36	931.15	41.49	808.59	41.68
	31_34	375.79	39.05	375.79	38.97	375.79	39.33	689.63	40.27	702.64	40.27	682.75	40.40	624.13	40.66
	33_36	291.44	38.14	291.44	38.06	291.44	38.38	544.47	39.26	548.12	39.22	540.53	39.37	486.29	39.60
106	21_24	2403.82	39.98	2403.82	40.53	2403.82	40.69	8053.32	42.66	8177.27	42.68	7777.66	42.71	5284.79	42.21
	23_26	1892.74	39.70	1892.74	40.18	1892.74	40.43	4741.05	42.21	4831.46	42.22	4592.44	42.28	3567.31	41.82
	25_28	1512.41	39.35	1512.41	39.72	1512.41	40.07	3164.94	41.79	3219.28	41.80	3058.16	41.88	2780.66	41.36
	27_30	1216.36	38.90	1216.36	39.18	1216.36	39.60	2417.74	41.28	2459.84	41.29	2326.43	41.39	2201.00	40.80
	29_32	980.38	38.36	980.38	38.54	980.38	39.02	1859.59	40.67	1892.95	40.68	1791.13	40.78	1820.25	40.16
	31_34	805.99	37.71	805.99	37.81	805.99	38.32	1452.44	39.91	1477.46	39.91	1395.04	40.01	1483.01	39.37
	33_36	637.83	36.96	637.83	36.98	637.83	37.49	1153.08	39.09	1169.15	39.05	1106.85	39.15	1206.19	38.49
175	21_24	1168.61	33.42	1168.61	36.24	1168.61	36.78	4318.03	42.66	4355.09	42.65	4184.42	42.66	2847.30	42.36
	23_26	860.49	33.31	860.49	35.90	860.49	36.59	2712.96	41.70	2740.25	41.68	2606.16	41.69	1904.85	41.51
	25_28	640.68	33.17	640.68	35.71	640.68	36.35	1800.22	40.76	1819.14	40.72	1721.13	40.75	1340.01	40.68
	27_30	480.79	32.99	480.79	35.14	480.79	36.00	1237.91	39.76	1256.05	39.73	1187.13	39.79	966.77	39.82
	29_32	364.14	32.77	364.14	34.72	364.14	35.59	869.80	38.73	881.30	38.69	833.51	38.81	713.36	38.89
	31_34	280.82	32.49	280.82	34.19	280.82	35.09	622.83	37.68	631.83	37.64	599.48	37.78	533.10	37.92
	33_36	215.76	32.12	215.76	33.35	215.76	34.33	456.36	36.55	462.34	36.51	442.66	36.66	408.95	36.89
185	21_24	3592.66	35.43	3592.66	36.21	3592.66	36.25	16000.08	40.56	15956.11	40.53	15894.71	40.54	10182.78	39.25
	23_26	2374.49	35.02	2374.49	35.67	2374.49	35.79	9895.20	39.26	9848.10	39.23	9780.20	39.26	6133.91	38.19
	25_28	1631.51	34.60	1631.51	35.14	1631.51	35.33	5702.23	38.09	5676.52	38.07	5600.60	38.11	3685.40	37.33
	27_30	1157.69	34.14	1157.69	34.55	1157.69	34.80	3284.06	37.05	3293.46	37.03	3232.74	37.08	2313.35	36.58
	29_32	841.25	33.59	841.25	33.87	841.25	34.15	1986.23	36.05	2008.86	36.04	1964.67	36.09	1541.83	35.83
	31_34	617.70	32.92	617.70	33.10	617.70	33.38	1281.85	35.01	1305.29	35.00	1269.60	35.07	1070.11	34.99
	33_36	459.32	32.15	459.32	32.23	459.32	32.50	886.40	33.91	907.41	33.91	885.06	34.00	769.85	34.04

Table 3.11: Codec integration PSNR results on PRIME7

VMAF	Combination	BL+DCT		BL+FRVSR		BL+DANet		BL+DCT+EL		BL+FRVSR+EL		BL+DANet+EL		SL	
VideoName	BL EL CRF	bitrate	VMAF	bitrate	VMAF	bitrate	VMAF	bitrate	VMAF	bitrate	VMAF	bitrate	VMAF	bitrate	VMAF
41	21_24	1240.47	90.43	1240.47	96.31	1240.47	97.41	2977.10	96.29	3005.82	96.42	2872.28	96.55	2262.15	96.20
	23_26	974.83	89.28	974.83	95.02	974.83	96.13	2185.20	95.38	2210.09	95.55	2094.52	95.79	1723.09	95.41
	25_28	775.18	87.78	775.18	93.31	775.18	94.53	1640.43	94.02	1657.98	94.38	1571.07	94.80	1331.35	94.22
	27_30	610.46	85.66	610.46	91.07	610.46	92.33	1246.95	92.34	1264.87	92.72	1199.69	93.31	1038.94	92.75
	29_32	480.87	82.96	480.87	88.17	480.87	89.46	943.16	89.93	958.81	90.54	911.22	91.27	815.39	90.66
	31_34	377.13	79.73	377.13	84.72	377.13	86.03	711.88	86.86	722.39	87.50	689.16	88.48	636.38	88.15
	33_36	295.40	75.62	295.40	80.34	295.40	81.71	545.20	83.02	553.58	84.02	534.74	84.90	504.90	84.57
59	21_24	2373.41	94.72	2373.41	95.75	2373.41	96.09	5661.54	98.77	5942.86	98.81	5564.20	98.97	4273.37	97.81
	23_26	1792.36	91.69	1792.36	92.73	1792.36	93.25	3970.26	97.25	4157.05	97.34	3894.01	97.66	3180.58	96.02
	25_28	1355.19	87.99	1355.19	88.95	1355.19	89.62	2810.95	94.49	2943.02	94.68	2761.03	95.11	2386.40	93.36
	27_30	1022.76	83.65	1022.76	84.56	1022.76	85.34	2053.17	90.79	2139.73	90.97	2017.09	91.47	1799.48	89.96
	29_32	777.77	78.65	777.77	79.47	777.77	80.32	1495.64	86.13	1554.91	86.35	1477.88	86.89	1358.75	85.87
	31_34	588.72	72.73	588.72	73.51	588.72	74.35	1088.72	80.44	1129.29	80.73	1080.23	81.30	1033.65	80.89
	33_36	448.37	66.12	448.37	66.82	448.37	67.71	822.49	73.89	849.30	74.30	818.69	74.79	788.29	74.95
71	21_24	1333.80	94.83	1333.80	96.49	1333.80	97.42	3132.16	97.50	3187.88	97.52	3048.88	97.74	2517.59	97.29
	23_26	1001.83	93.09	1001.83	94.87	1001.83	96.18	2013.13	96.38	2057.58	96.45	1944.32	96.87	1678.98	96.45
	25_28	764.51	90.65	764.51	92.39	764.51	93.98	1405.70	94.33	1441.72	94.54	1350.91	95.37	1211.90	94.86
	27_30	587.36	87.43	587.36	89.03	587.36	90.75	1023.82	91.27	1053.30	91.50	992.87	92.68	913.86	92.42
	29_32	453.80	83.36	453.80	84.80	453.80	86.61	754.24	87.04	777.69	87.41	735.40	88.89	707.44	89.07
	31_34	354.63	78.23	354.63	79.54	354.63	81.31	559.50	81.86	579.05	82.33	552.03	83.99	553.78	84.62
	33_36	275.45	71.97	275.45	73.11	275.45	74.86	425.46	75.39	439.23	75.84	422.51	77.61	438.15	78.96
87	21_24	1461.28	92.95	1461.28	95.08	1461.28	96.44	3425.77	97.09	3504.05	97.08	3413.47	97.32	2469.62	96.79
	23_26	1097.68	90.72	1097.68	92.90	1097.68	94.44	2378.72	95.71	2431.81	95.77	2364.33	96.16	1837.43	95.52
	25_28	837.98	88.06	837.98	90.22	837.98	91.83	1699.28	93.62	1737.24	93.66	1676.11	94.29	1384.72	93.70
	27_30	638.42	84.79	638.42	86.88	638.42	88.49	1271.51	90.72	1292.01	90.83	1256.50	91.42	1054.00	91.23
	29_32	489.18	80.76	489.18	82.79	489.18	84.37	944.23	86.91	954.39	87.18	931.15	87.97	808.59	88.09
	31_34	375.79	75.77	375.79	77.76	375.79	79.21	689.63	82.17	702.64	82.36	682.75	83.10	624.13	84.05
	33_36	291.44	70.08	291.44	72.02	291.44	73.41	544.47	76.80	548.12	76.87	540.53	77.70	486.29	79.29
106	21_24	2403.82	99.05	2403.82	99.17	2403.82	99.22	8053.32	99.31	8177.27	99.34	7777.66	99.33	5284.79	99.28
	23_26	1892.74	98.92	1892.74	99.04	1892.74	99.10	4741.05	99.19	4831.46	99.22	4592.44	99.21	3567.31	99.17
	25_28	1512.41	98.71	1512.41	98.90	1512.41	98.96	3164.94	99.06	3219.28	99.10	3058.16	99.11	2780.66	99.05
	27_30	1216.36	98.29	1216.36	98.49	1216.36	98.62	2417.74	98.89	2459.84	98.91	2326.43	98.90	2201.00	98.94
	29_32	980.38	97.73	980.38	97.96	980.38	98.16	1859.59	98.57	1892.95	98.63	1791.13	98.64	1820.25	98.64
	31_34	805.99	96.48	805.99	97.06	805.99	97.34	1452.44	97.92	1477.46	98.01	1395.04	98.09	1483.01	98.17
	33_36	637.83	93.80	637.83	95.15	637.83	96.04	1153.08	97.19	1169.15	97.31	1106.85	97.39	1206.19	97.35
175	21_24	1168.61	88.94	1168.61	98.01	1168.61	98.06	4318.03	97.56	4355.09	97.46	4184.42	97.58	2847.30	97.47
	23_26	860.49	88.10	860.49	97.22	860.49	97.44	2712.96	96.92	2740.25	96.93	2606.16	97.01	1904.85	96.84
	25_28	640.68	87.07	640.68	96.03	640.68	96.16	1800.22	96.00	1819.14	96.03	1721.13	96.24	1340.01	95.99
	27_30	480.79	85.28	480.79	94.28	480.79	94.57	1237.91	94.57	1256.05	94.82	1187.13	95.08	966.77	94.85
	29_32	364.14	83.19	364.14	91.80	364.14	92.31	869.80	92.55	881.30	93.04	833.51	93.11	713.36	93.08
	31_34	280.82	80.63	280.82	89.06	280.82	89.74	622.83	90.26	631.83	90.59	599.48	91.01	533.10	91.09
	33_36	215.76	77.24	215.76	84.91	215.76	85.77	456.36	86.78	462.34	87.35	442.66	87.72	408.95	88.07
185	21_24	3592.66	95.47	3592.66	97.83	3592.66	98.45	16000.08	99.31	15956.11	99.33	15894.71	99.38	10182.78	98.69
	23_26	2374.49	93.80	2374.49	96.47	2374.49	97.37	9895.20	98.82	9848.10	98.86	9780.20	98.93	6133.91	97.64
	25_28	1631.51	91.98	1631.51	94.86	1631.51	96.01	5702.23	97.78	5676.52	97.85	5600.60	98.01	3685.40	96.21
	27_30	1157.69	89.84	1157.69	92.74	1157.69	94.14	3284.06	95.95	3293.46	96.07	3232.74	96.36	2313.35	94.55
	29_32	841.25	86.97	841.25	89.77	841.25	91.30	1986.23	93.10	2008.86	93.30	1964.67	93.76	1541.83	92.31
	31_34	617.70	83.27	617.70	85.85	617.70	87.41	1281.85	89.20	1305.29	89.46	1269.60	90.18	1070.11	89.16
	33_36	459.32	78.42	459.32	80.78	459.32	82.34	886.40	84.29	907.41	84.51	885.06	85.49	769.85	85.06

Table 3.12: Codec integration VMAF results on PRIME7

PSNR	BL+DCT vs BL+DANet		BL+DCT+EL vs BL+DANet+EL		BL+DCT vs BL+ FRVSR		BL+DCT+EL vs BL+FRVSR+EL	
VideoName	BD-PSNR	BD-Rate (%)	BD-PSNR	BD-Rate (%)	BD-PSNR	BD-Rate (%)	BD-PSNR	BD-Rate (%)
41	1.42	-43.18	0.27	-6.14	1.28	-34.64	-0.05	1.16
59	0.13	-3.85	0.21	-6.20	-0.27	8.75	-0.15	4.62
71	0.25	-9.44	0.21	-7.13	-0.04	1.44	-0.05	1.82
87	0.33	-10.31	0.15	-4.28	-0.03	1.13	-0.06	1.66
106	0.68	-22.85	0.15	-8.04	0.28	-8.90	-0.02	1.34
175	2.97	-97.95	0.14	-5.16	2.19	-79.38	-0.06	2.39
185	0.64	-28.74	0.06	-2.41	0.44	-18.61	-0.03	1.25
Average	0.92	-30.90	0.17	-5.62	0.55	-18.60	-0.06	2.03

Table 3.13: Codec integration BD-rate and BD-PSNR results on PRIME7

VMAF	BL+DCT vs BL+DANet		BL+DCT+EL vs BL+DANet+EL		BL+DCT vs BL+ FRVSR		BL+DCT+EL vs BL+FRVSR+EL	
VideoName	BD-VMAF	BD-Rate (%)	BD-VMAF	BD-Rate (%)	BD-VMAF	BD-Rate (%)	BD-VMAF	BD-Rate (%)
41	6.60	-41.99	1.26	-15.58	5.36	-35.49	0.34	-4.25
59	1.61	-8.92	0.80	-5.93	0.90	-4.98	-0.31	2.48
71	3.14	-18.40	1.50	-12.64	1.56	-9.47	-0.09	0.88
87	3.62	-21.53	0.82	-7.23	2.08	-13.22	-0.06	0.55
106	0.53	-11.86	0.10	-7.88	0.34	-7.48	0.03	-2.32
175	9.13	-66.43	0.58	-11.59	8.77	-62.14	0.14	-2.96
185	3.95	-36.89	0.50	-9.25	2.71	-26.41	0.06	-1.08
Average	4.08	-29.43	0.79	-10.01	3.10	-22.74	0.02	-0.96

Table 3.14: Codec integration BD-rate and BD-VMAF results on PRIME7

4. SUMMARY AND CONCLUSIONS

4.1 Summary

Based on FRVSR we proposed DANet which runs VSR on the X265 compressed video, by adding the de-artifact module, our model performs 0.28dB and 0.81 VMAF better than the original model. The VSR results are used as inter-layer reference pictures in scalable video coding. X265 is used as the base layer codec whose motion vectors and quantization maps are imported to SHM for the benefits of enhancement layer coding. The proposed methods achieved -5.62% BD-rate reduction at the same video quality and 0.17 dB BD-PSNR quality improvement at the same bitrates than the original SHVC on PRIME7 test set.

We also evaluate the different options of downsampling filters and choose Lanczos as our downsampler for the sharpness of the base layer as it is required to be viewed separately from the enhancement layer.

Different options of processing VSR frames are also evaluated to reduce the buffering needed for VSR due to the discrepancy between encoding order and display order. The nested temporal layer method can remove the buffering with only 0.09 dB performance drop on VID4 test set.

4.2 Further Study

In the future, we have plans to do frame-level integration with nestedTID method which involves synchronization between the codec and VSR code base. Also, other information like the prediction residual from the BL encoder might help improve the de-artifact performance. The DANet might utilize that information to achieve even better performance. The reason why the subjective VMAF score is lower after EL encoding also needs an explanation.

REFERENCES

[1] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[2] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1646–1654, 2016.

[3] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 136–144, 2017.

[4] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690, 2017.

[5] M. S. Sajjadi, B. Scholkopf, and M. Hirsch, "Enhancenet: Single image super-resolution through automated texture synthesis," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4491–4500, 2017.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[7] Y. Romano, J. Isidoro, and P. Milanfar, "Raisr: rapid and accurate image super resolution," *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 110–125, 2016.

[8] Y. Jo, S. Wug Oh, J. Kang, and S. Joo Kim, "Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3224–3232, 2018.

[9] Y. Tian, Y. Zhang, Y. Fu, and C. Xu, "Tdan: Temporally-deformable alignment network for video super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3360–3369, 2020.

[10] X. Wang, K. C. Chan, K. Yu, C. Dong, and C. Change Loy, "Edvr: Video restoration with enhanced deformable convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.

[11] M. S. Sajjadi, R. Vemulapalli, and M. Brown, "Frame-recurrent video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6626–6634, 2018.

[12] D. Fuoli, S. Gu, and R. Timofte, "Efficient video super-resolution through recurrent latent space propagation," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 3476–3485, IEEE, 2019.

[13] M. Chu, Y. Xie, J. Mayer, L. Leal-Taixé, and N. Thuerey, "Learning temporal coherence via self-supervision for gan-based video generation," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 75–1, 2020.

[14] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 764–773, 2017.

[15] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," 2016.

[16] J. Chen, J. Boyce, Y. Ye, M. Hannuksela, and N. Barroux, "Jctvc-v1007: Shvc test model 11 (shm 11) introduction and encoder description," 02 2016.

[17] Y. H. J. Dong and Y. Ye, "Downsampling filter for anchor generation for scalable extensions of hevc," *m24499, 100th MPEG meeting, Geneva, CH*, 04 2012.

[18] ITU-T and ISO/IEC, "High efficiency video coding," *ITU-T and ISO/IECdocument ITU-T Rec. H.265 and ISO/IEC 23008–2 (HEVC)*, 11 2019.

[19] A. Mercat, M. Viitanen, and J. Vanne, "Uvg dataset: 50/120fps 4k sequences for video codec analysis and development," in *Proceedings of the 11th ACM Multimedia Systems Conference*, MMSys '20, (New York, NY, USA), p. 297âĂŞ302, Association for Computing Machinery, 2020.

[20] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *International Journal of Computer Vision (IJCV)*, vol. 127, no. 8, pp. 1106–1125, 2019.

[21] "Itu-t rec. p.910 (04/2008) subjective video quality assessment methods for multimedia applications," 2009.

[22] Z. Li, C. Bampis, J. Novak, A. Aaron, K. Swanson, A. Moorthy, and J. Cock, "Vmaf: The journey continues," *Netflix Technology Blog*, vol. 25, 2018.