



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

**Análise de qualidade do Low Complexity
Enhancement Video Coding (LCEVC) baseado em
AVC e VVC**

Lucas Araújo Pena

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Ricardo L. de Queiroz

Brasília
2025



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Análise de qualidade do Low Complexity Enhancement Video Coding (LCEVC) baseado em AVC e VVC

Lucas Araújo Pena

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Ricardo L. de Queiroz (Orientador)
CIC/UnB

Prof. Camilo Chang Dórea Prof. Edson Mintsu Hung
CIC/UnB ENE/UnB

Prof. Marcelo Grandi Mandelli
Coordenador do Bacharelado em Ciéncia da Computaçeo

Brasília, 30 de julho de 2025

Resumo

Este trabalho apresenta uma análise qualitativa do padrão de codificação de vídeo Low Complexity Enhancement Video Coding (LCEVC), padronizado pela Moving Picture Experts Group (MPEG) e desenvolvido pela V-Nova. O LCEVC atua como uma camada de aprimoramento sobre codecs existentes, oferecendo melhorias de qualidade com baixa complexidade computacional. Foram realizados testes com diferentes valores de quantização para a camada base (Advanced Video Coding / H.264 (AVC) e Versatile Video Coding / H.266 (VVC)) e dos parâmetros da camada de aprimoramento do LCEVC. Os resultados obtidos, analisados a partir da relação entre Peak Signal-to-Noise Ratio (PSNR) e a Taxa de bits, indicam que o LCEVC pode apresentar desempenho superior em certos cenários, especialmente em comparação ao uso isolado de codecs convencionais. A implementação dos testes foi realizada por meio de scripts automatizados em ambientes Linux. Este estudo reforça o potencial do LCEVC como uma solução eficiente para transmissão de vídeo de alta qualidade com baixo custo computacional.

Palavras-chave: LCEVC, AVC, VVC, compressão de vídeo, PSNR, análise de qualidade, TV Digital Brasileira

Abstract

This work presents a qualitative analysis of the Low Complexity Enhancement Video Coding (Low Complexity Enhancement Video Coding (LCEVC)) standard, standardized by the Moving Picture Experts Group (MPEG) and developed by V-Nova. LCEVC operates as an enhancement layer over existing codecs, providing quality improvements with low computational complexity. Tests were conducted with different quantization values for the base layer (Advanced Video Coding / H.264 (AVC) and Versatile Video Coding / H.266 (VVC)) and for the enhancement parameters of LCEVC. The obtained results, analyzed comparing Peak Signal-to-Noise Ratio (PSNR) and bitrate values, indicate that LCEVC can offer superior performance in certain scenarios, especially when compared to the isolated use of conventional codecs. The tests were implemented through automated scripts in Linux environments. This study highlights the potential of LCEVC as an efficient solution for high-quality video transmission with low computational cost.

Keywords: LCEVC, AVC, VVC, video compression, PSNR, quality analysis, Brazilian Digital TV

Sumário

1	Introdução	1
1.1	Motivação	1
1.2	Objetivo	1
1.3	Codificadores de vídeo	2
1.4	Advanced Video Coding / H.264 (AVC)	3
1.5	Versatile Video Coding / H.266 (VVC)	3
1.6	Taxa de bits e Peak Signal-to-Noise Ratio (PSNR)	4
1.6.1	Taxa de bits	4
1.6.2	Peak Signal-to-Noise Ratio (PSNR)	4
1.7	Low Complexity Enhancement Video Coding	5
1.7.1	Codificação	5
1.7.2	Decodificação	6
2	Metodologia	8
2.1	Codificando Vídeos	8
2.2	Análise dos Vídeos	8
2.3	Fronteira de Pareto	9
2.4	Sequências	10
2.5	Programas Usados	11
3	Resultados	12
3.1	AVC	13
3.1.1	Bosphorus	13
3.1.2	ReadySteadyGo	14
3.1.3	Jockey	15
3.1.4	SOCCER	16
3.1.5	City	17
3.1.6	vc-globo-05	18
3.1.7	vc-lcevc-01	19

3.1.8	vc-phillips-01	20
3.1.9	vc-phillips-03	21
3.2	VVC	22
3.2.1	Bosphorus	22
3.2.2	SOCCKER	23
3.2.3	Jockey	24
3.2.4	City	25
3.2.5	vc-phillips-01	26
3.2.6	vc-globo-05	27
3.3	Considerações	28
3.4	Visão geral dos resultados	29
4	Conclusão	30
4.1	Trabalhos Futuros	31
Referências		32
Anexo		33
I	Scripts	34
I.1	AVC	34
I.2	VVC	44
I.3	Resample	55

Lista de Figuras

1.1	Processo de codificação do LCEVC.	5
1.2	Processo de decodificação do LCEVC.	6
2.1	Exemplo da Fronteira de Pareto. [1]	10
2.2	Frames representativos das sequências de vídeo utilizadas.	11
3.1	Resultados para "Bosphorus" em AVC. [2]	13
3.2	Resultados para "ReadySteadyGo" em AVC. [2]	14
3.3	Resultados para "Jockey" em AVC. [2]	15
3.4	Resultados para "SOCCER" em AVC. [3]	16
3.5	Resultados para "City" em AVC. [3]	17
3.6	Resultados para "vc-globo-05" em AVC. [4]	18
3.7	Resultados para "vc-lcevc-01" em AVC.	19
3.8	Resultados para "vc-philips-01" em AVC.	20
3.9	Resultados para "vc-philips-03"em AVC.	21
3.10	Resultados para "Bosphorus" em VVC. [2]	22
3.11	Resultados para "SOCCER" em VVC. [3]	23
3.12	Resultados para "Jockey" em VVC. [2]	24
3.13	Resultados para "City" em VVC. [3]	25
3.14	Resultados para "vc-philips-01" em VVC.	26
3.15	Resultados para "vc-globo-05"em VVC [4].	27

Lista de Tabelas

1.1	Interpretação dos valores de PSNR. [5]	5
2.1	Versões dos programas utilizados para codificação e decodificação.	11
3.1	Compilado dos resultados finais para AVC.	29
3.2	Compilado dos resultados finais do VVC.	29

Lista de Abreviaturas e Siglas

AVC Advanced Video Coding / H.264.

DLSS Deep Learning Super Sampling.

EVC Essential Video Coding.

HEVC High Efficiency Video Coding / H.265.

ITU Telecommunication Standardization Sector.

JVET Joint Video Experts Team.

LCEVC Low Complexity Enhancement Video Coding.

MPEG Moving Picture Experts Group.

PSNR Peak Signal-to-Noise Ratio.

VVC Versatile Video Coding / H.266.

Capítulo 1

Introdução

1.1 Motivação

O crescimento do consumo de vídeo digital, impulsionado por serviços de streaming, transmissões ao vivo e a popularização de dispositivos conectados, trouxe desafios significativos para a transmissão de conteúdos em alta qualidade sem sobrecarregar as redes de comunicação. No contexto brasileiro, a demanda por vídeos em resoluções cada vez maiores, como 4K e 8K, exige soluções de compressão mais eficientes para garantir a entrega de conteúdo com qualidade e baixo custo operacional.

Nesse cenário, o Low Complexity Enhancement Video Coding (LCEVC) surge como uma tecnologia inovadora, capaz de aprimorar a eficiência de codificadores tradicionais ao adicionar camadas de aprimoramento com baixa complexidade computacional. Sua adoção tem ganhado destaque internacionalmente e, mais recentemente, no Brasil, onde foi escolhida como parte do padrão para a próximas gerações da TV Digital, parcialmente na TV 2.5 e por completo na TV 3.0 [6], onde atualmente é usado o codec Advanced Video Coding / H.264 (AVC). Essa escolha reflete o potencial do LCEVC em viabilizar transmissões de alta qualidade com menor uso de banda e recursos computacionais, tornando-se um tema relevante para pesquisa e análise no contexto nacional [7, 8].

1.2 Objetivo

Este trabalho tem como objetivo realizar uma análise qualitativa do desempenho do padrão Low Complexity Enhancement Video Coding (LCEVC), comparando seus resultados com os obtidos por codecs convencionais utilizados de forma isolada, como o AVC e o VVC. Para isso, são realizadas codificações das mesmas sequências de vídeo sob diferentes configurações de parâmetros, variando o fator de quantização da camada base e os valores de qualidade da camada de aprimoramento. Com base na relação entre PSNR e taxa de

bits, busca-se identificar em quais cenários o LCEVC apresenta ganhos reais de qualidade e eficiência.

1.3 Codificadores de vídeo

Quando se cria um vídeo para ser utilizado como mídia em alguma plataforma, é esperado que ele seja recebido e consumido com a melhor qualidade possível. Isso é um desafio que estas plataformas enfrentam há muitos anos. Por exemplo, quando se grava um filme, as filmagens são geradas em um formato bruto que ainda não sofreu nenhum tipo de edição ou compressão. Este formato gera vídeos de tamanhos gigantescos com a melhor qualidade possível [9]. Quando um filme está pronto para ser distribuído, é preciso fazer com que ele caiba em uma mídia física, como um DVD ou Blu-Ray, ou tenha um tamanho aceitável para ser transferido via streaming. Para que isso pudesse ser realizado, criou-se o padrão MPEG-2, que é capaz de codificar vídeos a taxas aproximadas de 4 a 9 Mb/s [10]. Esses algoritmos visam diminuir o tamanho de um vídeo, mas manter a melhor qualidade possível.

Com o aumento significativo do tráfego de dados em redes digitais e a popularização de serviços como streaming e videoconferências, a compressão eficiente de vídeo se tornou um fator crucial para garantir a qualidade destes serviços ao usuário final e também sua viabilidade técnica e econômica para os provedores destes serviços.

Apesar dos avanços destes *codecs*, os ganhos de compressão vieram acompanhados do aumento significativo da complexidade computacional, o que dificulta o seu uso em dispositivos com recursos limitados, como *smartphones*, receptores de TV ou plataformas web.

Segundo a Globo [11], serviços e aplicações envolvendo o uso de sinais de vídeo mantêm uma tendência de crescimento ao longo dos anos, onde o tráfego de vídeo responde por 80% do tráfego presente na Internet. Durante a pandemia, esse cenário incrementou sensivelmente, a ponto de alguns Governos Nacionais cogitarem a redução da oferta de sinais de vídeo de alguns serviços de streaming receando um possível colapso na infraestrutura de internet. Entende-se que a existência de codificadores mais eficientes é mais do que nunca uma forte demanda do mercado, oferecendo excelentes oportunidades para pesquisa e inovação.

Esses algoritmos foram sendo aprimorados com o passar do tempo e novos algoritmos foram sendo criados. Existem inúmeros algoritmos para compressão de áudio e vídeo, onde o foco agora serão os padrões Advanced Video Coding / H.264 (AVC) e Versatile Video Coding / H.266 (VVC), que serão utilizados neste trabalho.

1.4 Advanced Video Coding / H.264 (AVC)

O AVC é um padrão para compressão de vídeo baseado no MPEG-4 Parte 10, desenvolvido pela Telecommunication Standardization Sector (ITU) em conjunto com a MPEG. É um dos codificadores mais utilizados no mundo, devido a sua alta eficiência de compressão e boa qualidade de imagem, além da ampla compatibilidade.

O AVC é um codificador híbrido baseado em blocos, ou seja, ele divide o frame de um vídeo em pequenos blocos, onde nesses blocos são aplicadas as técnicas de compressão. Isso permite que o processamento seja segmentado facilitando a previsão, transformação e codificação. Ele emprega técnicas de compressão de movimento, transformada discreta do cosseno, quantização escalável e codificação de entropia. Ele possui um modelo de compressão temporal e compressão espacial. Estas características permitem a redução da taxa de bits sem degradação perceptível. Ele define uma arquitetura modular com níveis e perfis, para permitir sua aplicação em diferentes contextos, como streaming ou conteúdo de alta definição.

Este codificador se destaca em relação aos seus predecessores, como o MPEG-2, em eficiência de compressão. Essa característica foi essencial para que ele se tornasse um dos codificadores mais utilizados. [12]

1.5 Versatile Video Coding / H.266 (VVC)

O VVC também é um padrão de codificação de vídeo desenvolvido pela Joint Video Experts Team (JVET), que é uma colaboração entre o Telecommunication Standardization Sector (ITU) e o Moving Picture Experts Group (MPEG). Seu objetivo era dobrar a eficiência de compressão do codificador HEVC, mas mantendo a mesma qualidade visual. O VVC também é um codificador híbrido baseado em blocos, porém com diversas inovações que o tornam mais eficiente. Entre suas principais técnicas estão a unidade quadricomposta com partições adaptativas, melhorias no *intra-prediction*, maior sofisticação na compensação de movimento e filtros de interpolação, introdução de novos modos de transformadas e aprimoramento nas técnicas de codificação entropia.

Este codificador possui um amplo uso como vídeos com resoluções altas (como 8k), com realidade virtual, streaming e vídeo conferências em redes móveis. Possui suporte para HDR e um amplo espaço de cores. Porém, este alto desempenho vem como uma alta complexidade computacional, ainda maior que outros codificadores, especialmente na codificação, o que demanda otimizações específicas e o uso de aceleração por hardware, ou seja, o uso de placas de vídeo. [13]

1.6 Taxa de bits e Peak Signal-to-Noise Ratio (PSNR)

Para que fosse possível estudar, analisar e comprar sequências geradas, é necessário ter alguma métrica para poder compará-los. Para este caso, foi escolhido gerar um gráfico relacionando a taxa de bits e o PSNR de cada vídeo gerado dentro de um determinado parâmetro em comum.

1.6.1 Taxa de bits

A taxa de bits pode ser definida como o número de *bits* que é transferido ou processado por alguma unidade de tempo [14]. Este valor dá uma ideia da quantidade de recursos que serão necessários para transmitir e decodificar cada vídeo. Quanto maior este valor, maior foi o tempo necessário para a codificação, maior é o tamanho do arquivo, que acarreta em uma quantidade maior de dados sendo transferidos, caso for utilizado para streaming, e maior é a computação necessária para que o vídeo seja reconstruído pelo codificador. Então, este valor nos ajuda a determinar se um determinado ponto é ou não viável. Pode-se calcular a taxa de bits com a seguinte fórmula:

$$\text{Taxa de bits} = \frac{\text{Tamanho (em bits)}}{\text{Duração (em segundos)}}$$

1.6.2 Peak Signal-to-Noise Ratio (PSNR)

Signal-to-Noise Ratio, ou Relação Sinal-Ruído, é uma medida que compara a força de um sinal desejado com a força do ruído de fundo. É uma métrica utilizada para avaliar a qualidade e a confiabilidade de um sinal [14]. Para utilizar esta métrica na análise de vídeos, costuma-se usar o PSNR, que é a relação entre a máxima energia de um sinal e o ruído que afeta sua representação. Por muitos sinais terem sua amplitude dinâmica, o PSNR é normalmente expressado por uma escala logarítmica em decibéis.

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

Onde:

- PSNR: Peak Signal-to-Noise Ratio (em decibéis, dB);
- MAX: valor máximo possível de um pixel (por exemplo, 255 em 8 bits);
- MSE: erro quadrático médio entre os quadros original e comprimido;

Para este estudo, o importante é saber interpretar quanto cada valor representa na qualidade final do vídeo. Para isso, esta tabela demonstra a interpretação básica destes valores:

PSNR > 33 dB	Qualidade Excelente
33 dB > PSNR > 30 dB	Qualidade Aceitável
PSNR < 30 dB	Qualidade Ruim

Tabela 1.1: Interpretação dos valores de PSNR. [5]

1.7 Low Complexity Enhancement Video Coding

O LCEVC é uma tecnologia de codificação em múltiplas camadas desenvolvida para atuar como uma camada de aprimoramento sobre os codificadores já existentes, como os que foram mencionados anteriormente.

O LCEVC, por ser um codificador de vídeo, o LCEVC possui o processo de codificação e decodificação do vídeo, onde o decodificador é responsável por reconstruir o vídeo.

1.7.1 Codificação

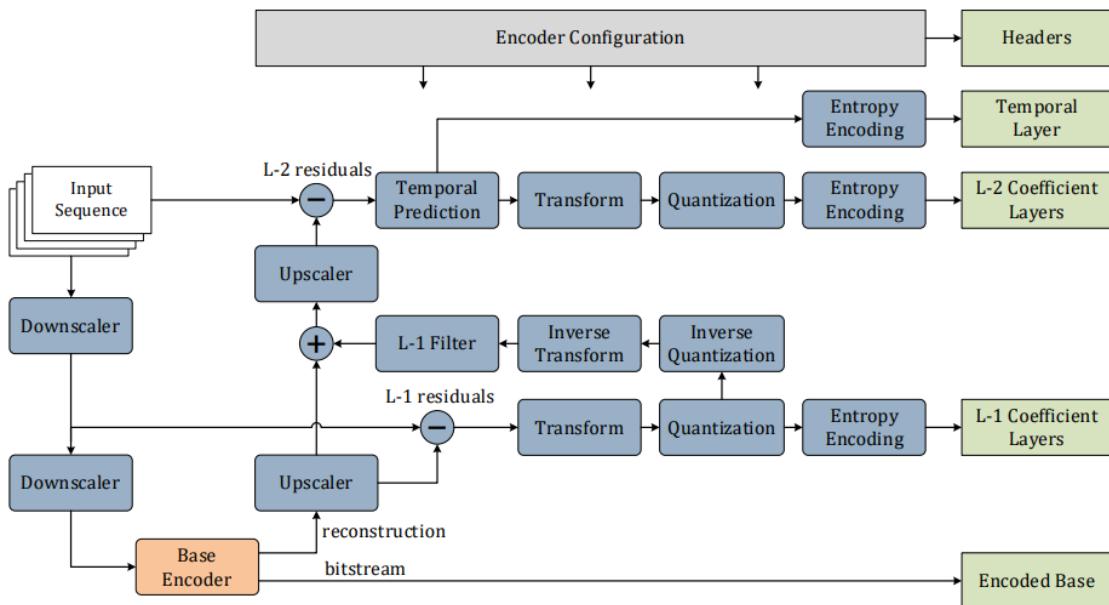


Figura 1.1: Processo de codificação do LCEVC (Fonte: [15]).

Camada Base

Primeiro, a sequência de entrada possui sua resolução reduzida por um processo chamado de *downscaling*. Ela passa por dois processos de *downscaling* que utilizam o parâmetro de modo de escalonamento (*scaling mode*) definido. Com o vídeo reduzido, então, é acionado o codificador responsável por codificar a camada base de acordo com o modo escolhido, que

pode ser AVC, VVC, HEVC ou EVC. O bitstream gerado pode ser incluído diretamente no fluxo LCEVC. [15, 16]

Subcamada de Aprimoramento 1 (L-1)

A imagem reconstruída da camada base é reamostrada para a resolução original e subtraída da versão redimensionada da sequência original. Essa diferença forma o *resíduo L-1*. Este é processado por ferramentas de codificação específicas (transformada, quantização e codificação de entropia), gerando coeficientes quantizados codificados. Esses dados compõem a primeira subcamada da camada de aprimoramento. [15, 16]

Subcamada de Aprimoramento 2 (L-2)

A reconstrução da subcamada L-1 é reconstruída internamente e, dependendo do modo de escalonamento, reamostrada novamente. A subtração da sequência de entrada original com essa reconstrução gera o *resíduo L-2*, que também é transformado e quantizado. Nesta etapa, é possível aplicar predição temporal aos coeficientes transformados, aumentando a eficiência de codificação. Os dados gerados, incluindo a informação de predição temporal por bloco, são inseridos no bitstream final. [15, 16]

1.7.2 Decodificação

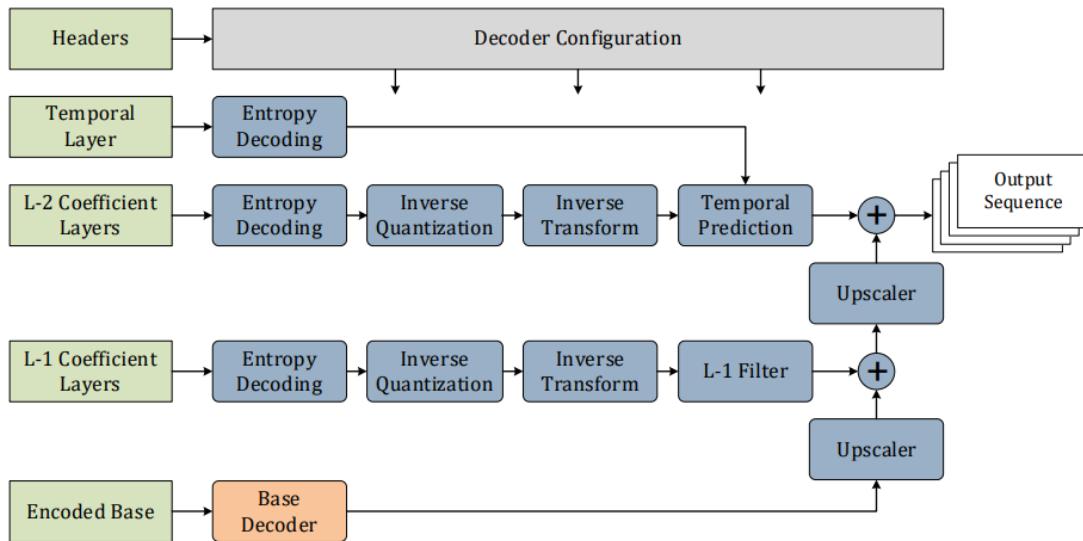


Figura 1.2: Processo de decodificação do LCEVC (Fonte: [15]).

O processo de decodificação de um vídeo com o LCEVC é composto por três etapas principais: decodificação da camada base, reconstrução da subcamada de aprimoramento 1 (L-1) e aplicação da subcamada de aprimoramento 2 (L-2). Abaixo, descrevemos cada uma dessas etapas.

Camada Base

O decodificador extrai e decodifica o bitstream da camada base utilizando o codec tradicional configurado. A imagem reconstruída, chamada de *Decoded Base Picture*, pode ser reamostrada (upscaleing) conforme o modo de escalonamento utilizado na codificação, formando a *Preliminary Intermediate Picture*. [15, 16]

Subcamada de Aprimoramento 1 (L-1)

Os coeficientes quantizados da subcamada [16]L-1 são decodificados utilizando as ferramentas inversas do processo de codificação (decodificação de entropia, desquantização e transformada inversa). Opcionalmente, é aplicado um filtro L-1 para suavizar as bordas dos blocos transformados. O resultado é somado à *Preliminary Intermediate Picture*, formando a *Combined Intermediate Picture*. Em seguida, pode ser aplicada uma nova etapa de upscaleing para obter a *Preliminary Output Picture*. [15, 16]

Subcamada de Aprimoramento 2 (L-2)

Por fim, a segunda subcamada de aprimoramento é decodificada. Caso os metadados indiquem o uso de predição temporal, os coeficientes são ajustados com base nos resíduos temporais do quadro anterior. Após a decodificação dos coeficientes, o resultado da subcamada L-2 é somado à *Preliminary Output Picture*, produzindo o quadro final decodificado, denominado *Combined Output Picture*. [15, 16]

Capítulo 2

Metodologia

2.1 Codificando Vídeos

Para codificar os vídeos para o LCEVC, foi usado o codificador LTM Model Encoder e Decoder, obtido através do Git da MPEG. Para codificar um vídeo utilizando este codificador, são passados os parâmetros pela linha de comando, ou através de um arquivo de configuração que possui os valores padrões do codificador. Como mencionado anteriormente, o LCEVC é constituído por duas camadas, a camada base contendo um vídeo com a metade da resolução do vídeo original, e uma camada de aprimoramento, que possui as informações para que o LCEVC possa realizar um upscaling no vídeo para que ele fique o mais parecido com o vídeo original possível. A camada base pode estar nos formatos AVC, VVC, HEVC e EVC. Já a camada de aprimoramento, possui duas subcamadas, onde uma é opcional e a outra é obrigatória, contendo os principais dados.

Para este trabalho, foi alterado o parâmetro que determina o nível da qualidade que será destinada para esta subcamada. Este valor determina a quantização desta camada, onde o quanto menor o valor, melhor a qualidade que esta camada possuirá, consequentemente aumentando o tamanho que a camada do LCEVC terá em relação ao arquivo final. Este valor é chamado de $SW2$, e seu parâmetro é `--cq_step_width_loq_0`. Além do valor de $SW2$ que modifica a qualidade da camada do LCEVC, foi modificado o valor do QP para a camada base. Para cada valor de QP , foi usado uma sequência de valores fixos para o $SW2$, para que fosse possível comparar a relação que estes valores iriam influenciar na qualidade e tamanho final do arquivo.

2.2 Análise dos Vídeos

Neste trabalho, será analisado o resultado de várias codificações utilizando o LCEVC, para tentar demonstrar qual seriam os melhores parâmetros para o seu uso, levando em

consideração os seus casos de uso. O programa utilizado foi compilado para Linux e sempre utilizado em uma distribuição Linux com base em Ubuntu.

Foi desenvolvido um *script Bash* para Linux [17] que executa o programa codificador paralelamente para cada valor de QP utilizado. O *script* cria uma execução por QP e processa sequencialmente por valor de $SW2$. Os valores utilizados foram:

QPs: 22, 25, 27, 30, 32, 35, 37.

SW2: 250, 750, 1250, 1750, 2250, 2750, 3250.

Além das codificações utilizando LCEVC, também foram codificados vídeos no mesmo padrão de compressão utilizado pelo LCEVC, mas sem o LCEVC, para que fosse possível ter uma referência da qualidade máxima possível obter sem o LCEVC. Neste caso, foram usados os mesmos valores de QP e utilizado a resolução final que o vídeo do LCEVC teria. Também foram extraídos os parâmetros que o LCEVC altera para o *downsampling* da camada base e adicionados no comando de execução da codificação sem o LCEVC.

Por fim, também é realizado uma codificação usando somente o algoritmo base de *Downsampling* e *Upsampling* que o codificador LCEVC utiliza. Porém, não há interferência de nenhum padrão de compressão neste estágio, onde ele transforma um YUV, um formato bruto de vídeo, em outro YUV. Por isso sua taxa de bits não é considerada.

Após cada vídeo ser codificado, o *script* também faz o cálculo de Peak Signal-to-Noise Ratio (PSNR) do resultado obtido, convertido para o formato YUV, assim como o arquivo original.

2.3 Fronteira de Pareto

A *Fronteira de Pareto* é uma técnica utilizada para analisar a relação entre dois ou mais objetivos, onde é possível identificar quais soluções são eficientes em relação a outros objetivos [18]. Neste trabalho, a Fronteira de Pareto é utilizada para analisar a relação entre a qualidade do vídeo, medida pelo PSNR, e o tamanho do arquivo, medido pela taxa de bits. A Fronteira de Pareto é representada por um conjunto de pontos que não podem ser melhorados em um objetivo sem piorar outro objetivo. Ou seja, se um ponto está na fronteira de Pareto, não é possível aumentar a qualidade do vídeo sem aumentar o tamanho do arquivo, ou diminuir o tamanho do arquivo sem diminuir a qualidade do vídeo.

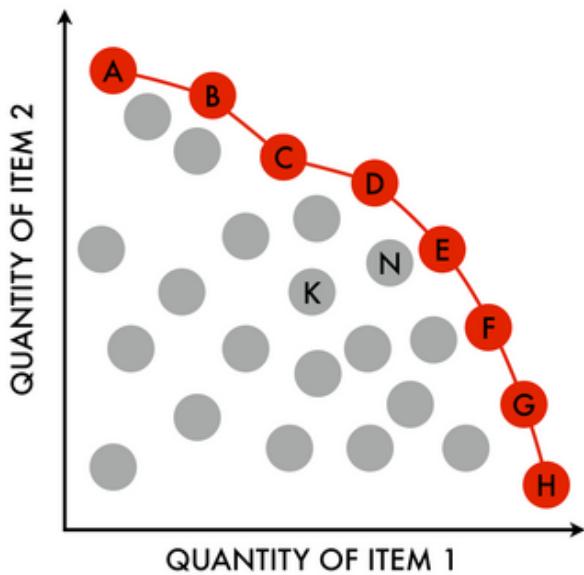


Figura 2.1: Exemplo da Fronteira de Pareto. [1]

2.4 Sequências

As sequências utilizadas para os testes deste trabalho foram obtidas em diversos locais diferentes, como: Ultra Video Group [2], Xiph.org [3] e vídeos que foram utilizados para testes da TV Digital, fornecidos pelo Laboratório de Engenharia Elétrica da UnB. A seguir uma breve descrição do conteúdo de cada sequência.

- Bosphorus: Um barco navegando pelo Estreito de Bósforo;
- Jockey: Um jóquei montado em seu cavalo correndo em um hipódromo. A câmera está com zoom e seguindo o cavalo.
- ReadySteadyGo: Uma largada de corrida de cavalos.
- vc-globo-05: Trecho de uma transmissão de futebol da Seleção Brasileira pela TV Globo.
- vc-lcevc-01: Um vídeo mostrando o horizonte e uma tabela fixa na tela.
- vc-phillips-01: Um campo em um dia claro. Plantas ao vento próximo da câmera.
- vc-phillips-03: Um vídeo de um canal em uma cidade.
- City: Uma visão aérea de Nova Iorque, focando em um prédio.
- SOCCER: Um trecho de um treinamento de futebol.



Figura 2.2: Frames representativos das sequências de vídeo utilizadas.

2.5 Programas Usados

Como mencionado previamente, o codificador e decodificador principal é o "*Test Model of Low Complexity Enhancement Video Coding*" LTM [19]. Quando é realizado a codificação do LCEVC com o formato escolhido da base, o LTM chama outro programa para codificar ou decodificar o arquivo. No caso deste trabalho, ao escolher o AVC, o LTM chama o JM, e no caso do VVC, o VTM é acionado. As versões utilizados estão na 2.1.

Programa	Versão
LTM (LCEVC Test Model)	7.0
JM (AVC Reference Software)	19.0
VTM (VVC Test Model)	12.0

Tabela 2.1: Versões dos programas utilizados para codificação e decodificação.

Capítulo 3

Resultados

Após a execução do Script de codificação, os dados obtidos são armazenados em um arquivo CSV. Com os resultados finais, outro script em *Python* é executado, onde ele cria um gráfico com estes dados. Abaixo estão os resultados obtidos.

A análise dos gráficos gerados para as sequências testadas permite observar o comportamento do LCEVC em diferentes condições de codificação. Foram analisados dois codificadores: AVC e VVC, com e sem o uso do LCEVC como camada de aprimoramento.

No geral, os gráficos PSNR x Taxa de bits permitem avaliar a eficiência da compressão, considerando que uma melhor relação é obtida quando se atinge maior qualidade (PSNR) com menor taxa de bits. A seguir, apresenta-se uma análise detalhada por cenário. Para alguns resultados, se um vídeo em LCEVC resultou em uma taxa de bits muito alta, ficando distante dos valores do codificador base, eles foram omitidos para melhor análise e visualização nos gráficos.

Além disso, para que o gráfico pudesse ficar legível, só foi rotulado o valor de SW2 para o parâmetro de QP37.

3.1 AVC

3.1.1 Bosphorus

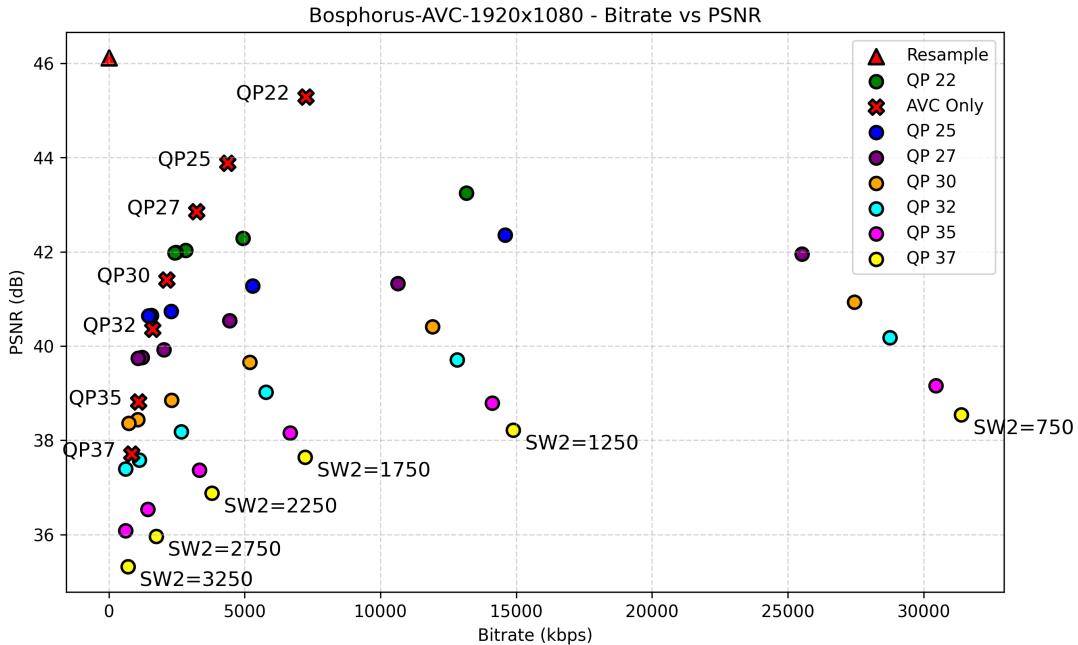


Figura 3.1: Resultados para "Bosphorus" em AVC. [2]

Para a sequência Bosphorus, os resultados revelam que o uso do LCEVC pode ser vantajoso em determinados cenários, dependendo dos parâmetros utilizados.

Nos testes com valores baixos de SW2, observou-se que, embora o PSNR alcançado seja alto, os arquivos resultantes apresentam uma taxa de bits superior ao necessário para atingir uma qualidade semelhante utilizando somente o AVC.

Para valores intermediários de SW2, os resultados foram mais promissores. Nestes casos, o LCEVC foi capaz de alcançar uma boa relação entre qualidade e taxa de bits. Nestes casos, embora o PSNR seja um pouco inferior, o LCEVC pode ser interessante em contextos onde a reconstrução visual seja mais importante que o valor do PSNR.

Para valores mais altos de SW2, como 2250 por exemplo, a camada de aprimoramento passou a contribuir muito pouco para a codificação, representando uma proporção bem pequena do tamanho total do arquivo. Neste caso, o LCEVC possui sua codificação comprometida pela perda de qualidade e espaço, tornando estas configurações pouco vantajosas.

3.1.2 ReadySteadyGo

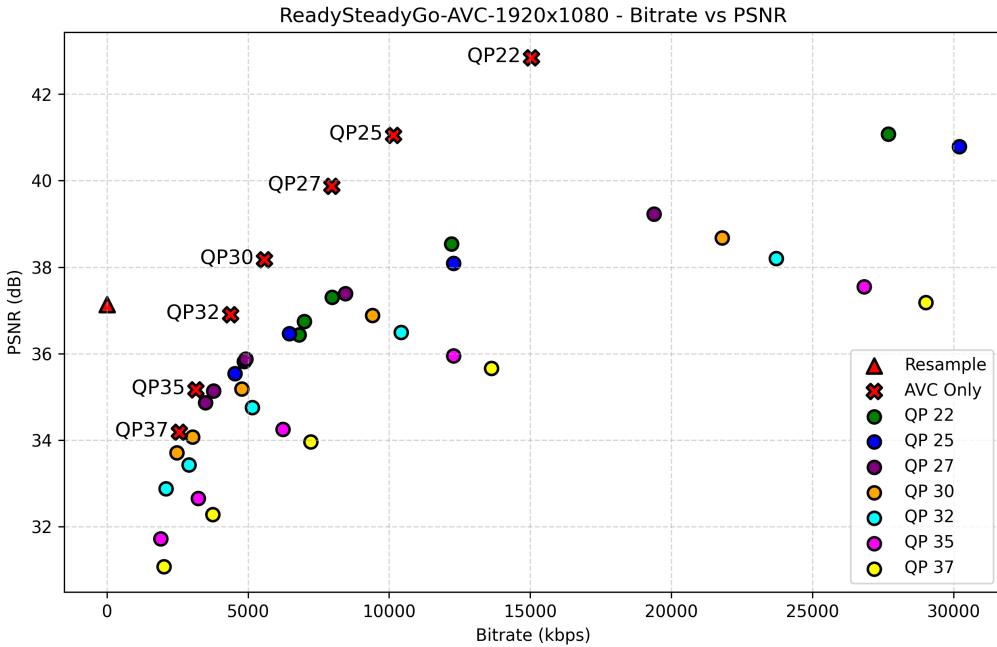


Figura 3.2: Resultados para "ReadySteadyGo" em AVC. [2]

Em geral, os resultados para esta sequência não demonstraram superioridade clara em comparação com o AVC puro.

Nos testes com o $SW2 = 1250$, os resultados demonstram taxas de bit significativamente mais altas, com ganhos pequenos ou mesmo nulos em PSNR. Mesmo em configurações com um QP menor, o LCEVC demonstrou uma taxa de bit muito alto, sem nenhuma melhoria de qualidade.

Para valores intermediários de $SW2$, houve uma redução na taxa de bit, mas também houve uma queda no PSNR.

Em valores maiores de $SW2$, a camada de aprimoramento passou a ter pouca ou nenhuma relevância, com taxas abaixo dos 10%. Com isso, os resultados se aproximam do desempenho da camada base, com valores baixos de PSNR, acarretando em uma perda significativa de qualidade.

Desta forma, esta sequência não se beneficiou com o uso do LCEVC, onde a relação entre PSNR e taxa de bits foi mais favorável ao AVC puro.

Vale destacar que o *Resample* do LCEVC obteve um PSNR relativamente baixo em relação aos resultados obtidos. Isto demonstra, que possivelmente, os resíduos que foram perdidos neste processo, foram compensados pela camada de aprimoramento

3.1.3 Jockey

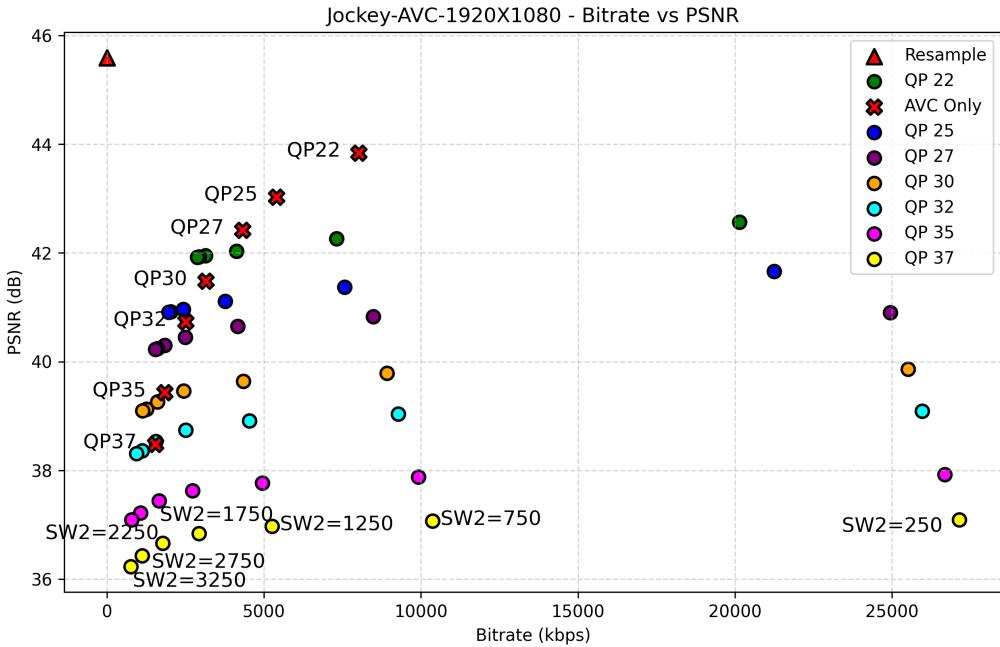


Figura 3.3: Resultados para "Jockey" em AVC. [2]

Para a sequência *Jockey*, o LCEVC alcançou ótimos resultados em relação às sequências de referência, onde há casos em que o LCEVC demonstrou um PSNR superior com uma taxa de bits menor. Percebe-se este comportamento em vários pontos antes do QP27 das sequências de referência. A partir desse ponto, os valores testados para o LCEVC decem e não são mais os melhores valores. Entretanto, considerando os resultados obtidos, é possível que com mais testes, surjam mais valores em que o LCEVC se saia melhor.

Aqui, vemos um caso em que o LCEVC se destacou e demonstrou vantagens em comparação ao AVC exclusivo, especialmente em configurações de taxas de bits intermediárias.

Em valores mais baixos, o LCEVC alcançou PSNR elevados com taxas de bits competitivas. Apesar da taxa mais alta, a relação qualidade-tamanho mostra que o LCEVC pode ser viável para aplicações que priorizam a qualidade visual.

Na faixa média, o LCEVC equilibrou melhor eficiência e qualidade, superando o AVC puro.

Para valores mais altos de SW2, a contribuição da camada de aprimoramento diminuiu, aproximando-se dos valores de somente AVC, mas ainda com ganhos moderados de PSNR.

Dessa maneira, o LCEVC é particularmente eficaz para conteúdos dinâmicos como "Jockey", onde a camada de aprimoramento compensa perdas da compressão base sem aumentar excessivamente a taxa de bit.

3.1.4 SOCCER

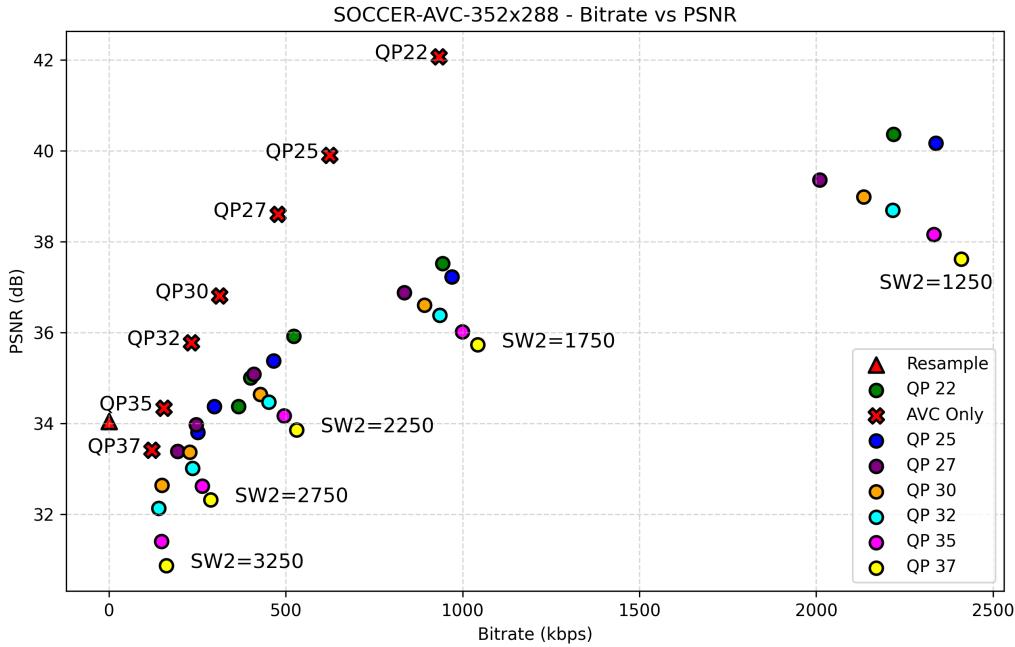


Figura 3.4: Resultados para "SOCCER" em AVC. [3]

Os resultados indicam que o LCEVC não apresentou vantagem clara em relação a somente o AVC na maioria das configurações testadas.

Em valores mais baixo, o LCEVC gerou arquivos com taxas de bits altas para pouco ganho em PSNR, enquanto o AVC puro atingiu qualidade similar com taxas de bits muito menores.

Nas faixas intermediárias de SW2, o LCEVC reduziu a taxa de bit, mas com perda acentuada de PSNR, ficando abaixo da curva de eficiência de somente o AVC.

Já em SW2 com valores maiores, a camada de aprimoramento se tornou irrelevante, resultando em desempenho próximo ao da camada base AVC, porém ainda inferior ao AVC puro em termos de PSNR.

O *Resample* do LCEVC também não demonstrou resultados significativos, com PSNR muito abaixo da maioria dos resultados obtidos.

Assim, o LCEVC não demonstrou um benefício em ser utilizado nesta sequência.

3.1.5 City

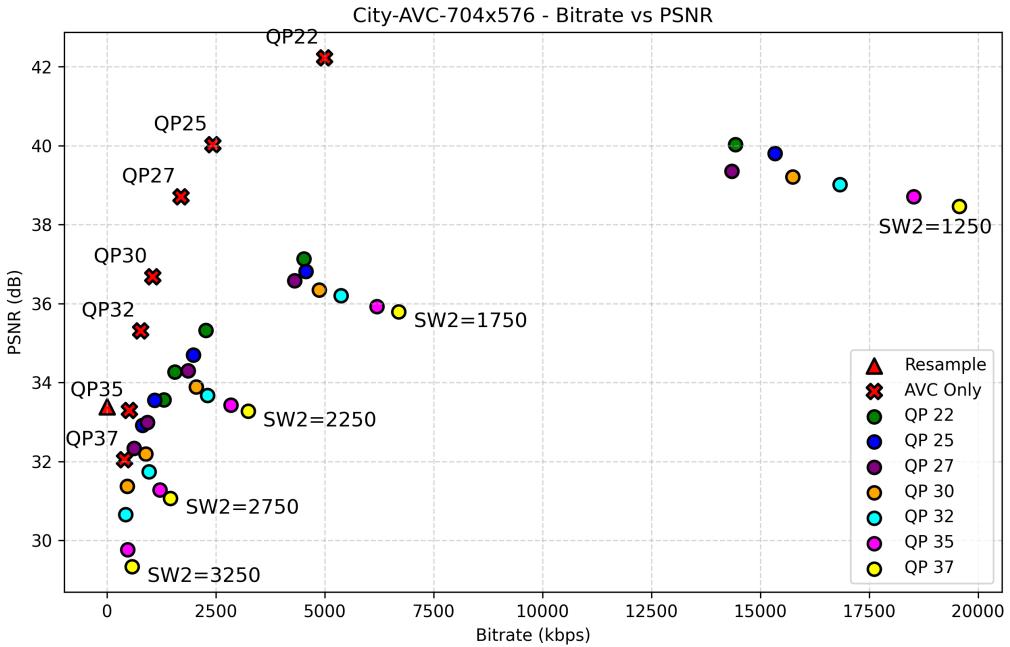


Figura 3.5: Resultados para "City" em AVC. [3]

Para esta sequência, os valores intermediários de SW2 proporcionam os resultados mais equilibrados. Nesses casos, o LCEVC foi capaz de gerar reconstruções visuais com ganhos notáveis em qualidade em relação ao AVC, porém com um ganho notável na taxa de bits.

Percebe-se que para valores iniciais de SW2, que é 250, os valores obtidos estão aproximadamente na curva que representa os melhores valores, porém, considerando o valor para AVC com QP37, ele possui a mesma taxa de bits de todos os pontos iniciais do LCEVC, porém com um PSNR superior.

Para valores maiores de SW2, o LCEVC demonstrou um ganho de qualidade, mas foi um ganho que não acompanhou o ganho em qualidade do AVC puro, além do aumento do tamanho da camada de aprimoramento, que acarretou no aumento da taxa de bits.

O *Resample* do LCEVC nesta sequência também não demonstrou um resultado como esperado, ficando com o PSNR semelhando ao QP35 do AVC puro, onde este ponto deveria possuir um PSNR superior aos valores obtidos para o AVC.

3.1.6 vc-globo-05

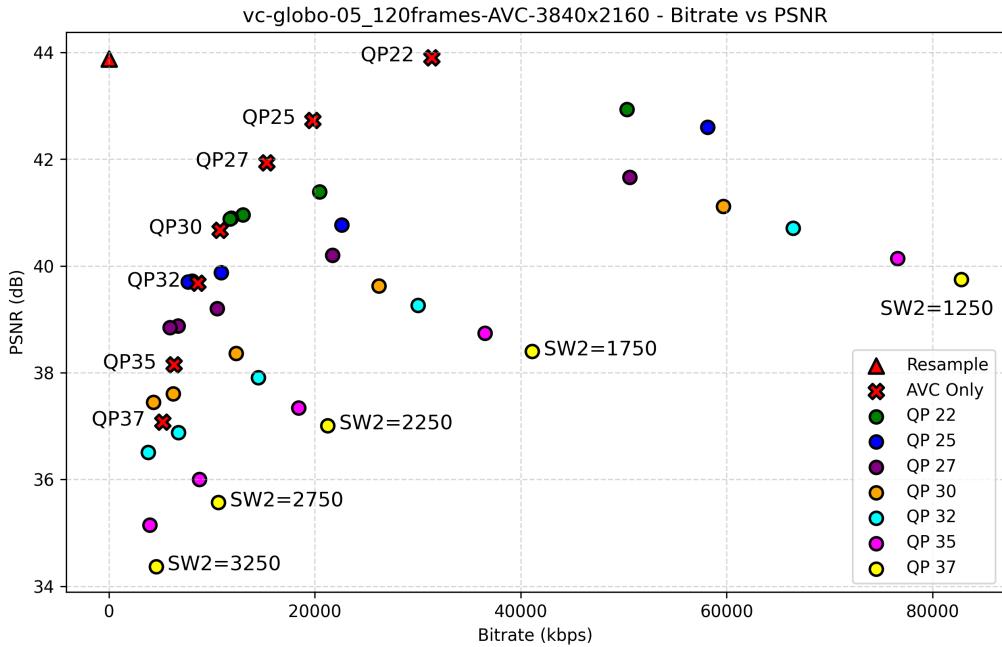


Figura 3.6: Resultados para "vc-globo-05" em AVC. [4]

Estes resultados mostram que para esta sequência, os resultados obtidos com LCEVC foram acima da média em comparação com outros resultados. A curva que resulta dos valores de SW2 para cada QP com o LCEVC está mais elevada, demonstrando que houve uma perda menor de PSNR e que a qualidade está próxima dos vídeos com somente AVC.

Para $SW2 = 1250$, os resultados possuem uma taxa de bits bastante elevada em comparação com a curva dos vídeos de referência. O que representa que o tamanho do arquivo gerado foi superior a todos os pontos à esquerda.

Os valores 3250 e 2750 de SW2 resultaram em valores de PSNR que estão acima da curva dos vídeos de referência, mostrando que o LCEVC conseguiu gerar resultados mais eficientes do que os vídeos que usaram somente o AVC.

Outro ponto a se observar é que o *Resample* do LCEVC obteve um valor de PSNR bastante próximo ao valor obtido pelo $QP = 22$ do AVC.

3.1.7 vc-lcevc-01

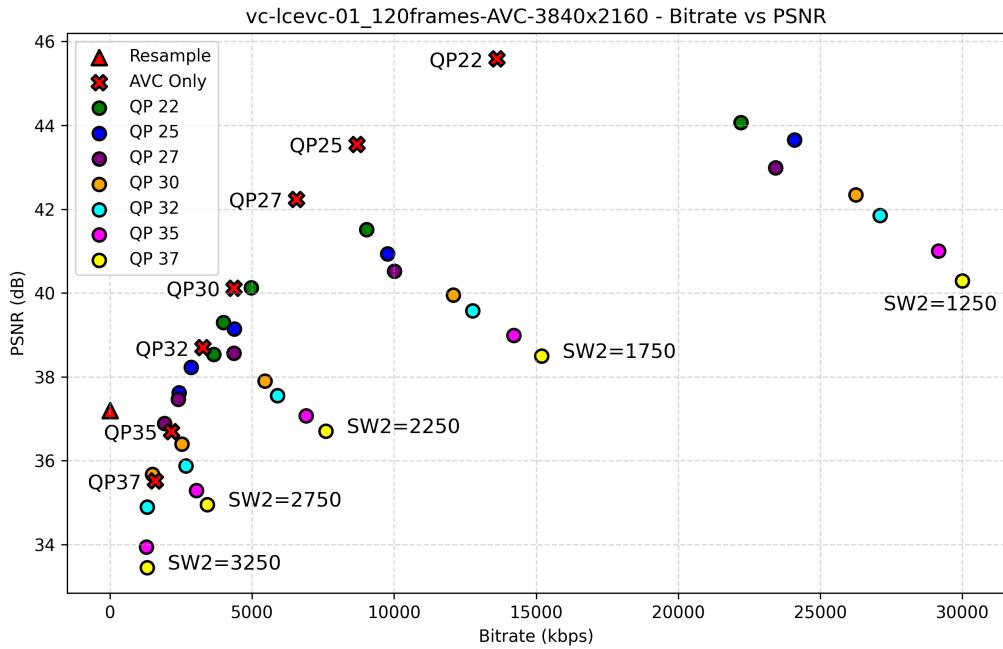


Figura 3.7: Resultados para "vc-lcevc-01" em AVC.

Aqui, com o $SW2 = 1250$, os dados mostraram que a camada de aprimoramento foi fortemente utilizada, com uma proporção de aprimoramento superior a 90% em todos os casos, resultando em uma taxa de bits elevada. Reduzindo para o valor 1750, houve uma melhora na eficiência.

Nos valores para $SW2$ iguais a 1750 e 2250, os resultados foram se aproximando da curva dos vídeos referência.

Agora, para $SW2$ com valores de 2750 e 3250, houve uma redução na taxa de bits. Nestes valores, os resultados foram próximos e até superior aos valores de PSNR dos vídeos somente em AVC.

3.1.8 vc-phillips-01

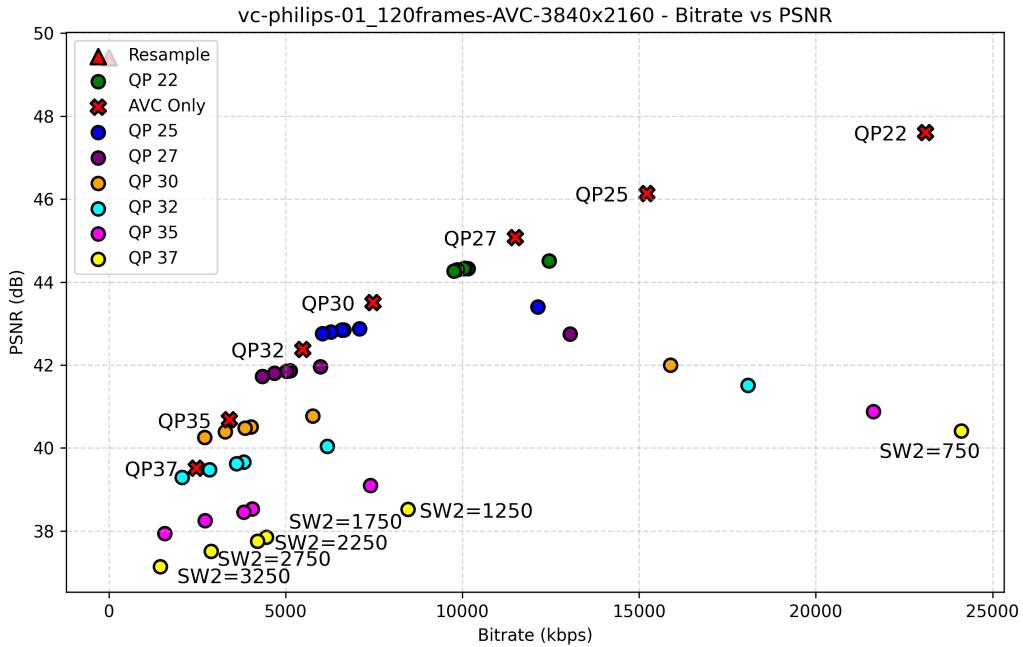


Figura 3.8: Resultados para "vc-phillips-01" em AVC.

A sequência "vc-phillips-01" apresentou uma característica interessante: mesmo com QPs elevados, os valores de PSNR se mantiveram altos, indicando que o conteúdo possui um baixo nível de ruído e variação espacial, o que favorece a compressão.

Nos testes com $SW2 = 750$, a proporção da camada de aprimoramento variou entre 84% e 97%. Para $QP = 30$, o LCEVC atingiu 42,00 dB a 15,9 Mbps, enquanto o AVC alcançou 43,51 dB com apenas 7,5 Mbps. Para valores como 1250 e 1750 para o $SW2$, a taxa de bits reduziu drasticamente para a maioria dos QPs, mantendo o PSNR alto.

Esta sequência foi bastante interessante, pois para valores de QPs iguais a 27, 25 e 22 para o LCEVC, os resultados foram eficientes, com uma taxa de bits semelhante ao do AVC, onde o LCEVC consegue demonstrar superioridade em alguns casos.

3.1.9 vc-phillips-03

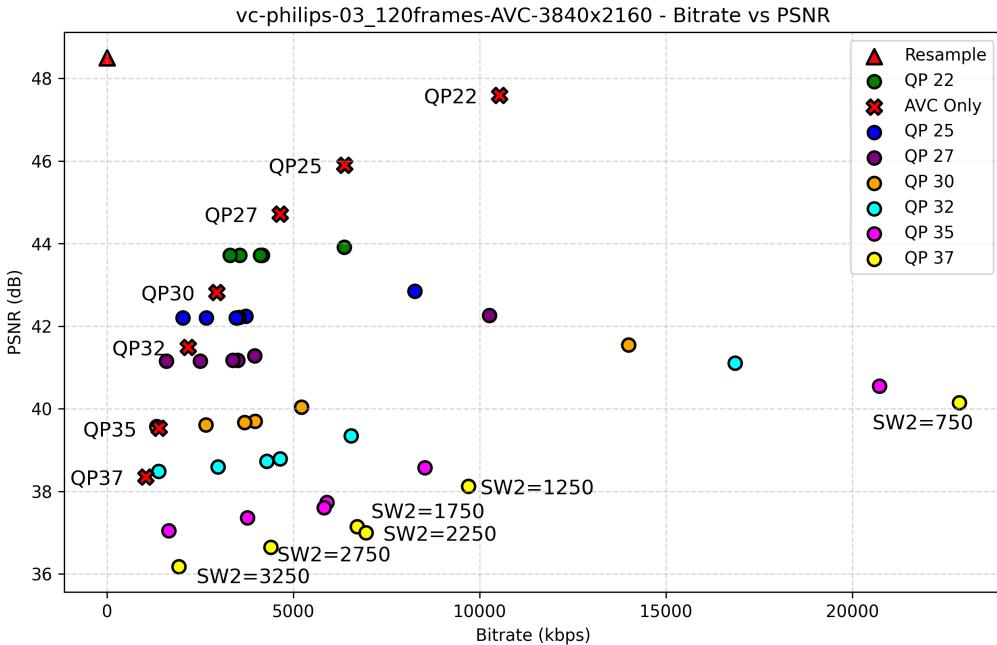


Figura 3.9: Resultados para "vc-phillips-03"em AVC.

Os resultados obtidos para esta sequência demonstraram um cenário promissor para o uso do LCEVC. Com alguns casos em que ele superou o AVC puro. Houve um grande aumento da taxa de bits para os resultados entre os valores de SW2 750 e 1250, onde o aumento foi proporcional ao valor do QP da base.

Vale destacar um dos resultados obtidos pela configuração de $SW2 = 1250$ e $QP = 22$, onde o LCEVC alcançou um PSNR de 43,72 dB com uma taxa de bits de apenas 4,1 Mbps, enquanto o AVC alcançou 47,59 dB a 10,5 Mbps. Mesmo com o PSNR do AVC sendo superior, a diferença de quase 2,5 vezes na taxa de bits pode justificar o uso do LCEVC em cenários com limitação de banda.

Em posições intermediárias de $SW2$, como 1750 e 2250, em QPs mais baixos, o LCEVC se aproximou da curva de eficiência do AVC, onde para o $QP = 22$, por exemplo, esteve bem próximo da taxa de bits do AVC.

Para o $SW2 = 750$, a taxa de bits foi muito alta, onde a camada de aprimoramento consumiu boa parte do tamanho do arquivo, acarretando em uma taxa de bits elevado para um PSNR relativamente baixo, que não escalou bem com o aumento do QP.

3.2 VVC

3.2.1 Bosphorus

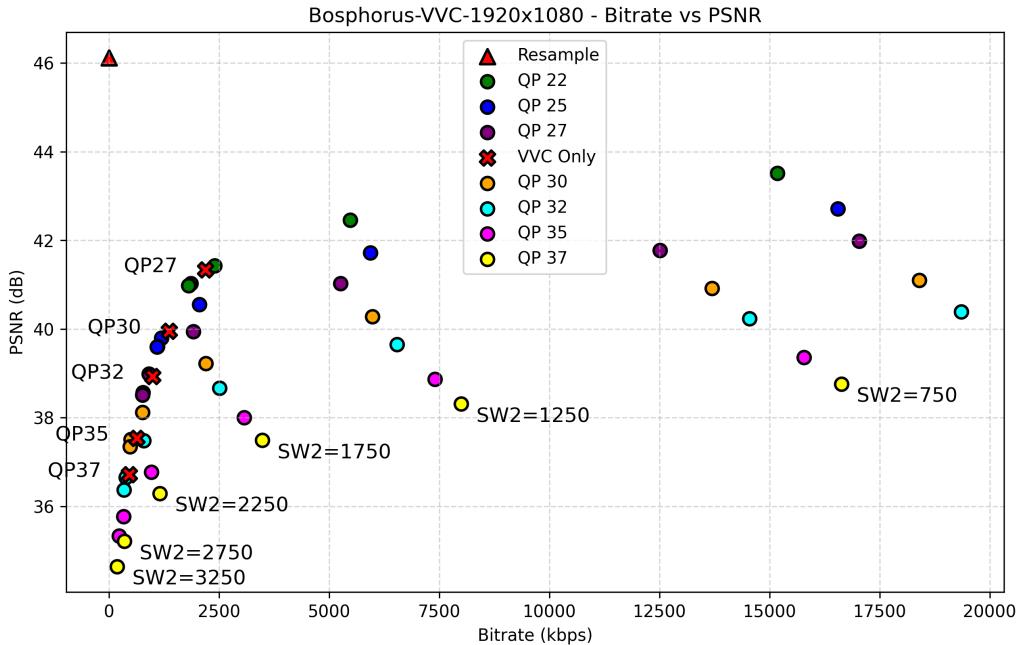


Figura 3.10: Resultados para "Bosphorus" em VVC. [2]

Com valores baixos de SW2, a camada de aprimoramento possui peso expressivo no tamanho final do arquivo, acarretando em uma taxa de bits elevado. Embora o ganho de qualidade seja pequeno, o aumento da taxa de bits é significativo, tornando estes resultados poucos eficientes.

Para valores intermediários, os valores começam a ter um equilíbrio melhor, se aproximando dos valores de somente VVC.

Agora, para valores de SW2 altos, como 3250 e 2750, começam a aparecer resultados que estão acima da curva gerada dos vídeos que utilizaram somente o VVC. Isso demonstra que para esta sequência, é possível utilizar o LCEVC e conseguir resultados praticamente iguais aos do VVC.

3.2.2 SOCCER

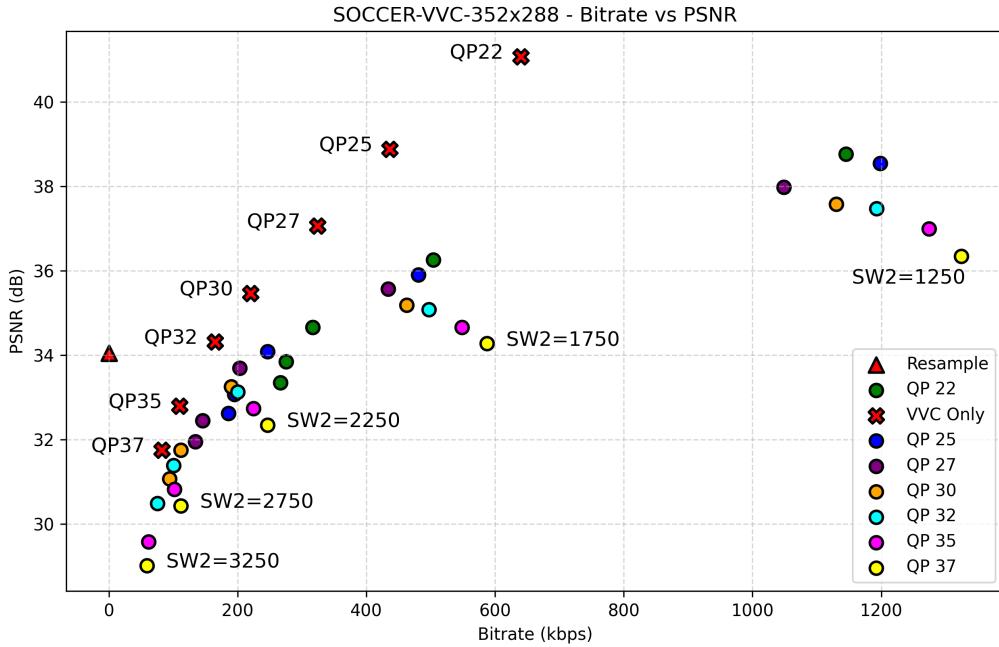


Figura 3.11: Resultados para "SOCCKER" em VVC. [3]

Os resultados demonstram que neste caso houve uma vantagem para o uso de somente o VVC, com desempenho superior na maioria das configurações avaliadas, principalmente em eficiência na relação qualidade e taxa. Aqui a eficiência do LCEVC foi abaixo do esperado, onde manteve uma relação alta muita das vezes, indicando baixa eficiência na codificação da camada base.

Nota-se que para valores de $SW2 = 1250$, os resultados demonstram uma taxa de bits maior que o $QP=22$ do VVC e um PSNR muito inferior, tornando estes resultados ineficientes.

O *Resample* desta sequência também demonstrou um resultado ruim, atingindo PSNR menos que o $QP=32$ do VVC.

Nesta sequência, o LCEVC não apresentou vantagens, e muitas vezes resultou no que seria uma perda de banda, sem ganhos significativos no PSNR.

3.2.3 Jockey

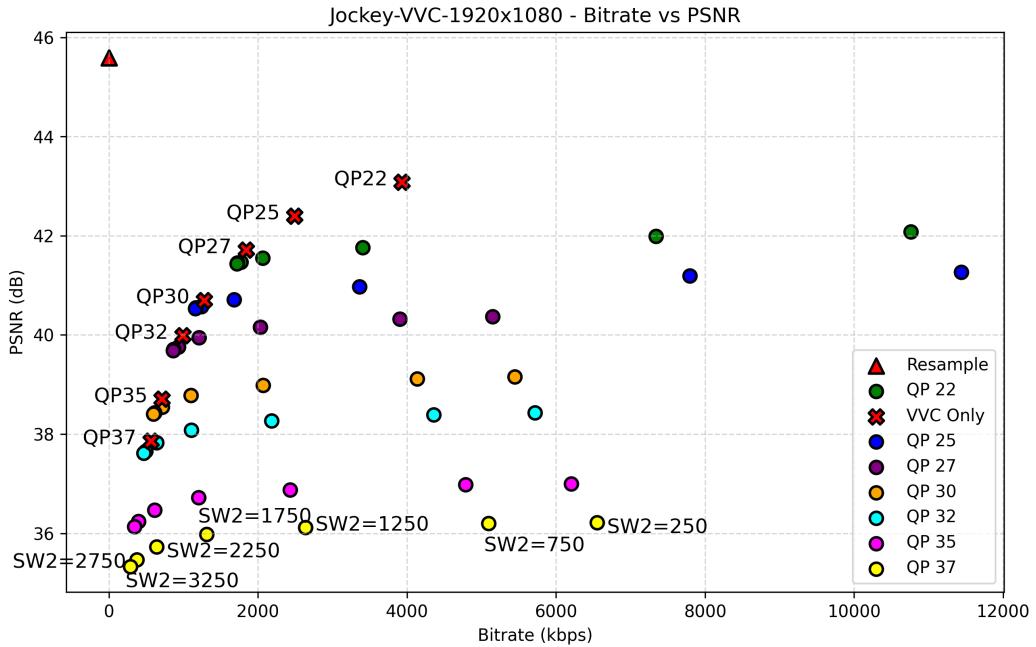


Figura 3.12: Resultados para "Jockey" em VVC. [2]

A sequência "Jockey" apresentou resultados bastante equilibrados entre o uso do VVC puro e o uso combinado do LCEVC. Avaliando os resultados obtidos, observa-se que o desempenho do LCEVC e da taxa de bits revela que o desempenho do LCEVC, neste caso, se aproxima consideravelmente do VVC isolado, e até melhor em alguns casos.

Para os resultados do LCEVC com QPs iguais a 22 e 25, observou-se um comportamento incomum, onde a taxa de bits destes pontos cresceu muito acima do esperado, saindo bastante da reta formada pelos pontos com o mesmo SW2.

Nesta sequência, a qualidade do LCEVC se manteve consistente e próxima dos valores obtidos por somente o uso do VVC. Isso torna o uso do LCEVC nesta sequência uma opção válida e demonstra ser uma alternativa válida para casos de uso com vídeos semelhantes.

3.2.4 City

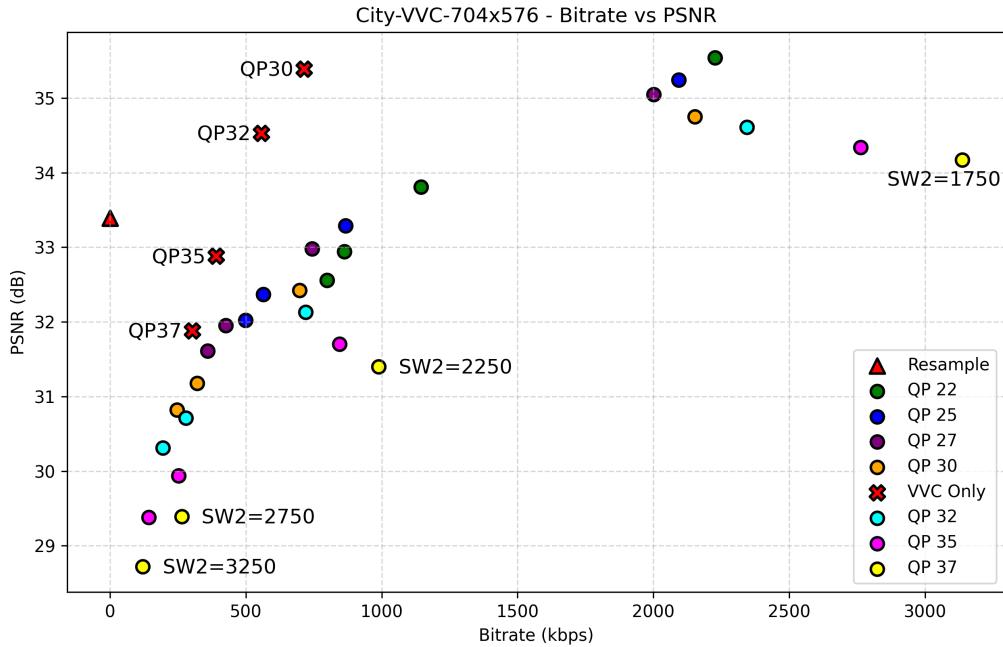


Figura 3.13: Resultados para "City" em VVC. [3]

Na comparação VVC puro e LCEVC + VVC, o desempenho do VVC isolado se mostrou eficiente. Esta sequência demonstra que o uso do VVC puro se mantém mais eficiente em praticamente todos os cenários testados.

Para o $SW2 = 1750$, os resultados do LCEVC apresentaram um valor elevado para a taxa de bits. Todos os estes resultados estão acima de uma taxa de bits de 2000 kbps, tornando estes resultados ineficientes em relação aos resto dos valores.

O LCEVC não trouxe ganhos práticos para esta sequência, onde somente o VVC por si só foi mais eficiente e apropriado para codificação de sequências como este vídeo aéreo de Nova York.

Outro ponto interessante deste resultado é que o *Resample* do LCEVC obteve uma qualidade inferior a vários pontos do VVC puro e do LCEVC + VVC.

3.2.5 vc-phillips-01

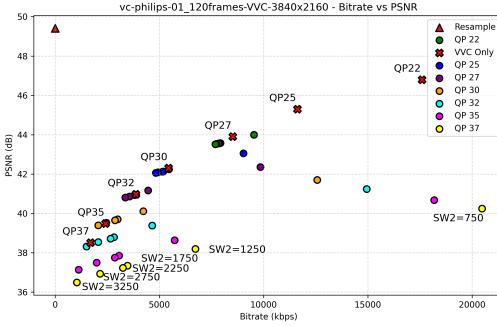


Figura 3.14: Resultados para "vc-phillips-01" em VVC.

Nesta sequência, o $SW2 = 750$ obteve uma taxa de bits bem alto em relação ao $SW2 = 1250$, onde também houve uma distância relativamente maior entre os valores de QP dentro do $SW2 = 750$.

Em $SW1 = 1250$, os pontos começaram a se aproximar da curva de eficiência, mas ainda assim com os valores abaixo da curva. O resto dos resultados para valores de $SW2$ menores ficaram bem próximos e tendendo ao formato da curva do VVC.

Conforme o valor de QP da base do LCEVC foi diminuindo, mais próximos os resultados para tal QP foram ficando. Os resultados de QP igual a 22, 25 e 27, ficaram muito próximos na maioria dos valores de $SW2$. Como as proporções da camada de aprimoramento destes valores ficaram entre 0,2% e 3,2%, isto afetou a eficiência do LCEVC, que ficou mais perceptível em uma sequência em 4K.

O LCEVC conseguiu alcançar ótimos resultados, onde vários pontos estiveram acima da curva de eficiência do VVC, entrando para a fronteira de Pareto.

3.2.6 vc-globo-05

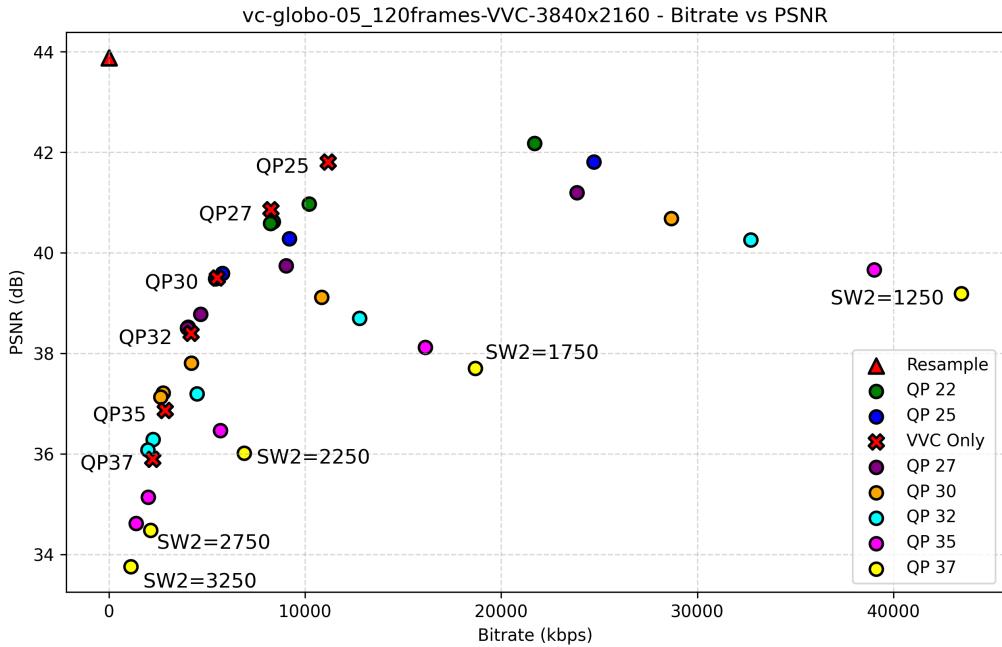


Figura 3.15: Resultados para "vc-globo-05" em VVC [4].

Os resultados os vídeos gerados com o LCEVC e VVC alcançaram um desempenho similar, onde há vários pontos na curva de melhor qualidade tanto do LCEVC quanto do VVC. Os resultados para valores mais baixos de SW2 continuam com uma diferença grande na taxa de bits entre eles.

No ponto QP = 25 e SW2 = 1250, o LCEVC alcançou um PSNR de 41,81 dB a 24,7 Mbps, exatamente o mesmo valor do VVC com QP = 25, porém com uma Taxa de bits maior que o sobre da solução base de somente o VVC, que obteve uma taxa de bits de 11,1 Mbps.

O resultados do LCEVC com QP = 32 e 30, foram ligeiramente superiores aos resultados do VVC, onde a taxa de bits foi menor e o PSNR foi superior. Na prática, esta vantagem é imperceptível, mas demonstra que o LCEVC pode ser vantajoso em alguns casos.

3.3 Considerações

- O comportamento do LCEVC varia conforme a sequência testada, o que é esperado, já que o padrão atua como uma camada adaptativa;
- O LCEVC demonstrou um desempenho superior em sequências que envolvam mais movimentação de câmera e detalhes, como no caso da sequência "Jockey";
- A vantagem do LCEVC se mostra mais evidente em cenários com a taxa de bits mais restrita, onde o refinamento da imagem se torna crucial;
- Em taxas mais altas, o benefício da camada de aprimoramento tende a diminuir, pois a camada base já está oferecendo uma boa qualidade;
- Também é importante observar que o uso do LCEVC com codecs mais simples, como o AVC tende a ser mais vantajoso do que com outros codecs mais avançados, como o VVC, uma vez que o ganho sobre algo já bem otimizado costuma ser menor;
- Nos testes realizados, foi possível observar que mesmo quando há uma proporção bem alta para a camada de aprimoramento, o ganho de PSNR não é direto, onde muitos casos a taxa de bits do LCEVC é muito maior que um vídeo utilizando somente o codificador base, e a qualidade aferida pelo PSNR é muito menor.

3.4 Visão geral dos resultados

Após a análise de cada sequência, foi feito uma interpretação dos resultados e definido qual deles se saiu melhor no geral. Vale ressaltar que mesmo que um tipo de codificação tenha se saído melhor, como a maioria dos resultados foram muito próximos, pode não haver uma diferença significativa na qualidade visual ou tamanho. Foram considerados somente os pontos que estão próximos da curva de eficiência (Fronteira de Pareto).

Sequência	Resultado
Bosphorus (1920x1080, AVC)	Leve vantagem para o LCEVC
ReadySteadyGo (1920x1080, AVC)	Vantagem para o AVC
Jockey (1920x1080, AVC)	Vantagem para o LCEVC
SOCCKER (352x288, AVC)	Vantagem para o AVC
City (704x576, AVC)	Vantagem para o AVC
vc-globo-05 (3840x2160, AVC)	Empate
vc-lcevc-01 (3840x2160, AVC)	Empate
vc-phillips-01 (3840x2160, AVC)	Empate
vc-phillips-03 (3840x2160, AVC)	Leve vantagem para o LCEVC

Tabela 3.1: Compilado dos resultados finais para AVC.

Sequência	Resultado
Bosphorus (1920x1080, VVC)	Empate
SOCCKER (352x288, VVC)	Vantagem para o VVC
Jockey (1920x1080, VVC)	Empate
City (704x576, VVC)	Empate
vc-globo-05 (3840x2160, VVC)	Leve vantagem para o LCEVC
vc-lcevc-01 (3840x2160, VVC)	Vantagem VVC
vc-phillips-01 (3840x2160, VVC)	Leve vantagem para o LCEVC

Tabela 3.2: Compilado dos resultados finais do VVC.

Capítulo 4

Conclusão

Este trabalho apresentou uma análise qualitativa do padrão Low Complexity Enhancement Video Coding (LCEVC), investigando seu desempenho quando utilizado como camada de aprimoramento sobre codecs tradicionais, como AVC e VVC. A proposta do LCEVC de oferecer melhorias de qualidade com baixa complexidade é bastante útil para o contexto atual dos codificadores de vídeo.

Os teste realizados evidenciaram que o desempenho do LCEVC é fortemente atrelado ao tipo de conteúdo codificado e à parametrização aplicada, especialmente os valores de QP para a camada base e o SW2 para a camada de aprimoramento. De maneira geral, o LCEVC demonstrou resultados positivos em algumas condições específicas, particularmente em conteúdos com maior movimentação e complexidade temporal, como a sequência "Jockey", onde a camada de aprimoramento se mostrou eficaz na preservação da qualidade virtual, com uma taxa de bits eficiente.

Em outras sequências como "Soccer" e "City" os resultados indicam que o isolado dos codecs foi mais eficiente. O uso do LCEVC com uma baixa quantização (SW2) resultou em taxas de bits elevadas, sem ganhos de PSNR.

Os resultados indicam que a principal vantagem do LCEVC está na sua flexibilidade de adaptação ao cenário. Quando ele é bem parametrizado, ele oferece uma relação qualidade e taxa de bits competitiva, tornando-o uma solução atrativa para ambientes com restrições de hardware, banda ou complexidade computacional, onde ele permite que seja alcançado níveis satisfatórios de qualidade mesmo quando não é viável utilizar codecs mais exigentes. Outro ponto forte é sua alta customização de parâmetros, que o torna uma tecnologia com alto nível de adaptação a vários cenários.

Porém, seu uso requer cuidado nas escolhas dos parâmetros, porém, seus ganhos em eficiência nem sempre são garantidos, principalmente quando se utiliza codecs modernos, como observado com o VVC.

4.1 Trabalhos Futuros

Embora este trabalho tenha explorado o LCEVC em combinação com codecs tradicionais, há várias direções futuras que podem ser investigadas:

- Explorar o uso do LCEVC com outros codecs, como HEVC e EVC, para avaliar seu desempenho em diferentes cenários de compressão.
- Investigar a aplicação do LCEVC em contextos de transmissão ao vivo, onde a latência e a eficiência são críticas.
- Analisar o impacto do LCEVC em dispositivos com recursos limitados, como smartphones e dispositivos embarcados, para entender melhor sua viabilidade em cenários de baixa potência.
- Realizar uma análise mais aprofundada da complexidade computacional do LCEVC, comparando-a com outros métodos de aprimoramento de vídeo.
- Investigar a integração do LCEVC com técnicas de aprendizado de máquina para otimização de parâmetros e melhoria da qualidade do vídeo.
- Realizar um estudo mais profundo sobre os parâmetros de codificação do LCEVC, como o SW1 e outras centenas de parâmetros disponíveis, para entender melhor como eles afetam a qualidade e a eficiência da codificação.
- Explorar a possibilidade de utilizar o LCEVC em conjunto com técnicas de super-resolução para melhorar ainda mais a qualidade do vídeo em resoluções mais baixas. Como por exemplo, o uso do Deep Learning Super Sampling (DLSS) ao invés do algoritmo de *Upsampling* do LCEVC.

Referências

- [1] Wikipedia: *Pareto efficient frontier*. https://en.wikipedia.org/wiki/Pareto_front, acesso em 2025-07-27. vii, 10
- [2] Mercat, A., M. Viitanen e J. Vanne: *Uvg dataset: 50/120fps 4k sequences for video codec analysis and development*. Proc. ACM Multimedia Syst. Conf., June 2020. vii, 10, 13, 14, 15, 22, 24
- [3] Xiph.Org: *Xiph.org video test media*. <https://media.xiph.org/video/derf/>, acesso em 2025-05-05. vii, 10, 16, 17, 23, 25
- [4] TV Globo: *Trecho de vídeo da tv globo em formato raw/mov*, 2024. Conteúdo obtido para testes de compressão. Sem distribuição pública. vii, 18, 27
- [5] Syahbana, Yoanda Alim, Herman, Azizah Abdul Rahman e Kamalrulnizam Abu Bakar: *Aligned-psnr (apsnr) for objective video quality measurement (vqm) in video stream over wireless and mobile network*. Em *2011 World Congress on Information and Communication Technologies (WICT)*, Johor, Malaysia, 2011. IEEE. <https://core.ac.uk/download/pdf/228026495.pdf>. viii, 5
- [6] SBTVD: *Resultados da avaliação subjetiva da qualidade da codificação de vídeo em tempo real para tv 2.5.* 1
- [7] Carlos Eduardo Cosme Ribeiro: *Gt tv 3.0: Funcionalidades do mpeg-5, low complexity enhancement video coding (lcevc)*. <https://set.org.br/news-revista-da-set/revista/gt-tv-3-0-funcionalidades-do-mpeg-5-low-complexity-enhancement-video-coding-lcevc/>. 1
- [8] V-Nova: *Brazilian tv giant achieves 10mbps uhd with lcevc enhancement*. <https://v-nova.com/press-releases/brazilian-tv-giant-achieves-10mbps-uhd-with-lcevc-enhancement/>, acesso em 2024-09-05. 1
- [9] Miller, Jeremy: *What is raw footage and when to use it*. <https://bxfilms.tv/blog/what-is-raw-footage>, acesso em 2022-09-09. 2
- [10] Morris, O.J.: *Mpeg-2: where did it come from and what is it?* Em *IEE Colloquium on MPEG-2 - What it is and What it isn't*, páginas 1/1–1/5, 1995. 2
- [11] Carlos Eduardo Cosme Ribeiro e Globo: *Panorama dos codificadores de vídeos*. <https://globotech.globo.com/blog/noticia/panorama-dos-codificadores-de-videos.ghtml>, acesso em 2024-08-13. 2

- [12] ITU-T Study Group 16 (VCEG) and ISO/IEC JTC1/SC29/WG11 (MPEG): *ITU-T Recommendation H.264 / ISO/IEC 14496-10: Advanced video coding for generic audiovisual services*. Recommendation / international standard H.264 / ISO/IEC 14496-10, International Telecommunication Union, International Organization for Standardization International Electrotechnical Commission, maio 2003. <https://www.videosurveillance.co.in/H.264.pdf>, Corrigendum 1 added May 7, 2004. 3
- [13] Hamidouche, Wassim, Thibaud Biatek, Mohsen Abdoli, Edouard François, Fernando Pescador, Miloš Radosavljević, Daniel Menard e Mickael Raulet: *Versatile video coding standard: A review from coding tools to consumers deployment*. arXiv preprint, junho 2021. <https://arxiv.org/pdf/2106.14245>, Revised version (v2) submitted on 6 Nov 2021. 3
- [14] Gupta, Prakash C.: *Data Communications and Computer Networks*. PHI Learning Pvt. Ltd., New Delhi, India, 2006, ISBN 9788120328464. https://books.google.com.br/books?id=-kNn_p6WA38C&pg=PA7. 4
- [15] ISO/IEC JTC 1/SC 29/WG 11 MPEG-5 Part 2 Working Group: *White paper on low complexity enhancement video coding (lcevc)*. White paper MPEG-137, LCEVC (Low Complexity Enhancement Video Coding), janeiro 2022. 5, 6, 7
- [16] Battista, Stefano, Guido Meardi, Simone Ferrara, Lorenzo Ciccarelli, Florian Maurer, Massimo Conti e Simone Orcioni: *Overview of the low complexity enhancement video coding (lcevc) standard*. IEEE Transactions on Circuits and Systems for Video Technology, 32(11):7983–7995, 2022. 6, 7
- [17] Lucas Pena: *Lcevc-enhancement*. https://github.com/lucpena/LCEVC-Enhancement/tree/main/Low_Complexity_Enhancement_Video_Coding, acesso em 2025-07-27. 9, 34
- [18] Emmerich, Michael e André Deutz: *Multicriteria optimization and decision making: Principles, algorithms and case studies*, 2025. <https://arxiv.org/abs/2407.00359>, Página 26. 9
- [19] *LTM Git (requer uma conta MPEG)*. <https://git.mpeg.expert/MPEG/Video/LCEVC/LTM>. 11

Anexo I

Scripts

Todos os códigos criados para este trabalho estão disponíveis no GitHub [17]. Recomenda-se o GitHub para a análise do código, devido a organização dos arquivos. Para garantir a preservação destes códigos, eles serão anexados abaixo.

Os scripts foram divididos em duas baterias de codificação diferentes. Uma para AVC e uma para VVC. Cada um possui seu *main* que irá chamar todos os códigos necessários de maneira paralela, para agilizar a geração dos resultados. Entretanto, este script foi executado em um processador com vários núcleos que se beneficia com a paralelização. Caso for executar este script em um processador com menos recursos, recomenda-se a alteração do script para que ele seja executado sequencialmente, ou diminuir a quantidade de processos criados.

O script cria uma pasta chamada *avc_only*, onde é necessário colar nesta pasta o *encoder.cfg* do codificar AVC. Também é possível alterar o script e passar o endereço do *encoder.cfg*.

Coloque todos estes arquivos na mesma pasta.

I.1 AVC

main-avc.sh

```
#!/bin/bash

echo -en "\nInicializando Codificacao em AVC... "
for i in {5..1}; do
    sleep 1
    echo -n "${i} "
done
echo -e "\n"
```

```

# === Configuracoes compartilhadas ===
# InputFile="/mnt/md0/lucpena/videos/Bosphorus_1920x1080_120fps_420_8bit.yuv"
# InputFile="/mnt/md0/lucpena/videos/SOCCER_352x288_30_orig_02.yuv"
# InputFile="/mnt/md0/lucpena/videos/RaceNight_3840x2160_50fps_8bit.yuv"
# InputFile="/mnt/md0/lucpena/videos/ReadySteadyGo_1920x1080_120fps_420_8bit.
yuv"
# InputFile="/mnt/md0/lucpena/videos/Jockey_1920x1080_120fps_420_8bit.yuv"
# InputFile="/mnt/md0/lucpena/videos/city_704x576_yuv420p_60fps_600frames.yuv"
InputFile="/mnt/md0/lucpena/videos/SBTVD/YUV/vc-globo-05_120frames_420p.yuv"

# VIDEO_NAME="Bosphorus"
# VIDEO_NAME="SOCCER"
# VIDEO_NAME="RaceNight"
# VIDEO_NAME="ReadySteadyGo"
# VIDEO_NAME="Jockey"
# VIDEO_NAME="City"
VIDEO_NAME="vc-globo-05_120frames"

# WIDTH=1920
# HEIGHT=1080
# FPS=60
# NUM_FRAMES=600

## Soocer
# WIDTH=352
# HEIGHT=288
# FPS=30
# NUM_FRAMES=300

## City
# WIDTH=704
# HEIGHT=576
# FPS=60
# NUM_FRAMES=600

## SBTVD
WIDTH=3840
HEIGHT=2160
FPS=60

```

```

NUM_FRAME=120

SW2=250
SW2_max=3250
FORMAT="yuv420p"

BaseEncoder="avc"
Step=500

MAIN_TIME=$(date +"%Y-%m-%d_%H-%M-%S")

QP_list=(22 25 27 30 32 35 37)

ENCODER_SCRIPT="avc_template.sh"
SCRIPT_DIR=$(dirname "$(readlink -f "$0")")

BaseEncoderUpper=$(echo "$BaseEncoder" | tr '[lower:]' '[upper:]')
SizesFile="$SCRIPT_DIR/${VIDEO_NAME}-${BaseEncoderUpper}-${WIDTH}x${HEIGHT}-${
MAIN_TIME}.csv"

# Loop para LCEVC
for QP in "${QP_list[@]}"; do
    DIR="QP${QP}"
    mkdir -p "$DIR"
    cp "$ENCODER_SCRIPT" "$DIR/run.sh"

    # Cria script com todas as variaveis embutidas
    cat > "$DIR/config.sh" <<EOF
#!/bin/bash

MAIN_TIME="$MAIN_TIME"
InputFile="$InputFile"
VIDEO_NAME="$VIDEO_NAME"
WIDTH=$WIDTH
HEIGHT=$HEIGHT
FORMAT="$FORMAT"
FPS=$FPS
NUM_FRAME=$NUM_FRAME
SW2=$SW2
SW2_max=$SW2_max
Step=$Step

```

```

BaseEncoder="$BaseEncoder"
QP=$QP
BaseEncoderUpper="$BaseEncoderUpper"
SizesFile="$SizesFile"
EOF

# Cria o SizesFile se nao existir
if [ ! -f "$SizesFile" ]; then
    echo "file,sw2,qp,lv_size,base_size,enhance_size,enhancement_ratio,
psnr_avg,bitrate_kbps" > "$SizesFile"
fi

chmod +x "$DIR/run.sh"
chmod +x "$DIR/config.sh"

# Executa em segundo plano
(cd "$DIR" && ./run.sh) &
sleep 0.5
done

# === Script so com AVC ===
AVC_ONLY_DIR="$SCRIPT_DIR/avc_only"
AVC_TEMPLATE="$SCRIPT_DIR/avc_only_template.sh"
mkdir -p "$AVC_ONLY_DIR"

for QP in "${QP_list[@]}"; do
    QPDIR="$AVC_ONLY_DIR/QP${QP}"
    mkdir -p "$QPDIR"

    # Copia template e torna executavel
    cp "$AVC_TEMPLATE" "$QPDIR/run.sh"
    chmod +x "$QPDIR/run.sh"

    # Config file
    cat > "$QPDIR/config.sh" <<EOF
#!/bin/bash
MAIN_TIME="$MAIN_TIME"
InputFile="$InputFile"
VIDEO_NAME="$VIDEO_NAME"
WIDTH=$WIDTH

```

```

HEIGHT=$HEIGHT
FORMAT="$FORMAT"
FPS=$FPS
NUM_FRAMES=$NUM_FRAMES
BaseEncoder="$BaseEncoder"
QP=$QP
SizesFile="$SizesFile"
EOF
chmod +x "$QPDIR/config.sh"

# Dispara em segundo plano
(cd "$QPDIR" && ./run.sh "$QP") &
sleep 0.5
done

# === Script com Down/Up sampling ===
DIR="resample"
mkdir -p "$DIR"
cp "resample_template.sh" "$DIR/run.sh"

cat > "$DIR/config.sh" <<EOF
#!/bin/bash
MAIN_TIME="$MAIN_TIME"
InputFile="$InputFile"
VIDEO_NAME="$VIDEO_NAME"
WIDTH=$WIDTH
HEIGHT=$HEIGHT
FORMAT="$FORMAT"
FPS=$FPS
NUM_FRAMES=$NUM_FRAMES
SizesFile="$SizesFile"
EOF
chmod +x "$DIR/*.sh

(cd "$DIR" && ./run.sh) &

wait
echo -e "\n-----\n Finish \n-----\n"

```

avc_only_template.sh

```
#!/bin/bash

source ./config.sh

QP="$1"
if [[ -z "$QP" ]]; then
    echo "Erro: QP nao fornecido!"
    exit 1
fi

BaseEncoderUpper=$(echo "$BaseEncoder" | tr '[lower:]' '[upper:]')
SCRIPT_PATH=$(readlink -f "$0")
SCRIPT_DIR=$(dirname "$SCRIPT_PATH")
STATUS_FILE="$SCRIPT_DIR/status-${QP}.log"
ENC_DIR="/mnt/md0/lucpena/output/main/output"

rm -f "$STATUS_FILE"
echo -e "Iniciando o script AVC Only\n\n" >> "$STATUS_FILE"

SizesFileAVC="$SCRIPT_DIR/${VIDEO_NAME}-${BaseEncoderUpper}-only.csv"

# Checa se o CSV ja existe, se nao, cria com o cabecalho
if [ ! -f "$SizesFileAVC" ]; then
    echo "file,sw2,qp,lvc_size,base_size,enhance_size,enhancement_ratio,
psnr_avg,bitrate_kbps" > "$SizesFileAVC"
fi

JMEncoder="/home/lucpena/apps/LTM/_build_linux/external_codecs/JM/lencod"
FFMPEG="/home/lucpena/apps/FFMPEG-VMAF/ffmpeg"

TIMESTAMP=$(date +"%Y-%m-%d_%H-%M-%S")
OutputFile="$ENC_DIR/${VIDEO_NAME}_${BaseEncoderUpper}-Only_QP${QP}.264"
OutputFileName="${OutputFile##*/}"
OutputYUVFile="${OutputFile%.264}.yuv"
LogFile="$SCRIPT_DIR/${VIDEO_NAME}_${BaseEncoderUpper}-Only_QP${QP}.log"
CONFIG="/mnt/md0/lucpena/output/main/all/avc_only/encoder.cfg"

echo -e "\n==== Codificando somente AVC QP=${QP} ===\n" | tee -a "$STATUS_FILE"
```

```

echo "> Codificando com QP=$QP" >> "$STATUS_FILE"

# Codificando para AVC
${JMEncoder} -d "$CONFIG" -p InputFile="$InputFile" -p OutputFile="$OutputFile"
-p SourceWidth=$WIDTH -p SourceHeight=$HEIGHT -p FrameRate=$FPS -p
FramesToBeEncoded=$NUM_FRAMES -p YUVFormat=1 -p QPISlice=$QP -p QPPSlice=$((QP + 1)) -p QPBSlice=$((QP + 1)) -p IntraPeriod=64 2>&1 | tee "$LogFile"

# Convertendo o arquivo de saida para YUV para o PSNR
$FFMPEG -i "$OutputFile" -pix_fmt "$FORMAT" -f rawvideo "$OutputYUVFile" -y >>
"$STATUS_FILE"

# Calculando PSNR
$FFMPEG -s:v "${WIDTH}x${HEIGHT}" -pix_fmt "$FORMAT" -i "$InputFile" \
-s:v "${WIDTH}x${HEIGHT}" -pix_fmt "$FORMAT" -i "$OutputYUVFile" \
-lavfi psnr="stats_file=psnr.log" -f null - >> "$STATUS_FILE"

psnr_avg=$(tail -n 1 psnr.log | awk -F'psnr_avg:' '{print $2}' | awk '{print $1}')
rm psnr.log >> "$STATUS_FILE"

output_file_size=$(stat -c%s "$OutputFile")
bitrate_kbps=$(awk "BEGIN { printf \"%.\.2f\", ($output_file_size * 8 * $FPS) / ($NUM_FRAMES * 1000) }")

[[ -z "$psnr_avg" ]] && psnr_avg="N/A"

# Salva no CSV principal e no local
echo "${OutputFileName},264,$QP,$output_file_size,0,0,0%,${psnr_avg}, ${bitrate_kbps}" >> "$SizesFileAVC"
echo "${OutputFileName},264,$QP,$output_file_size,0,0,0%,${psnr_avg}, ${bitrate_kbps}" >> "$SizesFile"

echo "QP=$QP | PSNR=${psnr_avg} dB | Bitrate=${bitrate_kbps} kbps" >> "$STATUS_FILE"

echo -e "\n\n AVC Only finalizado!" | tee -a "$STATUS_FILE"

```

avc_template.sh

```
#!/bin/bash
source ./config.sh

SCRIPT_PATH=$(readlink -f "$0")
SCRIPT_DIR=$(dirname "$SCRIPT_PATH")

ModelEncoder="/home/lucpena/apps/LTM/_build_linux/ModelEncoder"
ModelDecoder="/home/lucpena/apps/LTM/_build_linux/ModelDecoder"
FFMPEG="/home/lucpena/apps/FFMPEG-VMAF/ffmpeg"

MAIN_DIR="/mnt/md0/lucpena/output/main"
BASES_DIR="$MAIN_DIR/bases"
DEC_DIR="$MAIN_DIR/dec"
LOGS_DIR="$MAIN_DIR/logs"
BIN_DIR="$MAIN_DIR/bin"
RESULTS_DIR="$MAIN_DIR/results"
STATUS_FILE="$SCRIPT_DIR/status.log"

LOCAL_CSV="$SCRIPT_DIR/${VIDEO_NAME}-qp${QP}.csv"

# Checa se o CSV ja existe, se nao, cria com o cabecalho
if [ ! -f "$LOCAL_CSV" ]; then
    echo "file,sw2,qp,lvc_size,base_size,enhance_size,enhancement_ratio,
          psnr_avg,bitrate_kbps" > "$LOCAL_CSV"
fi

rm -f "$STATUS_FILE"
echo -e "\n==== Iniciando testes para QP=${QP} ===\n" | tee -a "$STATUS_FILE"

rm -f $SCRIPT_DIR/*.yuv $SCRIPT_DIR/psnr.log 2>> "$STATUS_FILE"

BASE_NAME="${VIDEO_NAME}-qp${QP}-qp${BaseEnconderUpper}.bin"
if [[ ! -f "${BIN_DIR}/${BASE_NAME}" ]]; then
    echo -e "\nO arquivo ${BASE_NAME} nao existe, criando um novo..." | tee -a
    "$STATUS_FILE"
    ${ModelEncoder} \
        --width=${WIDTH} --height=${HEIGHT} --format=${FORMAT} --fps=${FPS}
        --qp=${QP} \
```

```

--input_file="${InputFile}" \
--output_file="$DEC_DIR/base.lvc" --base_encoder=${BaseEncoder}
--encapsulation=nal \
--cq_step_width_loq_0=${SW2} --keep_base \
2>&1 | tee -a "$LOGS_DIR/${VIDEO_NAME}_base-gen_QP${QP}_${(date +"%Y-%m-%d_%H-%M-%S")}.log"

mv "$SCRIPT_DIR/base.bin" "${BIN_DIR}/${BASE_NAME}"

echo -e "Base ${BASE_NAME} criada com sucesso." | tee -a "$STATUS_FILE"
else
    echo -e "Base ${BASE_NAME} ja existente, reutilizando." | tee -a "$STATUS_FILE"
fi

for (( current_sw2=SW2; current_sw2<=SW2_max; current_sw2+=Step )); do
    TIMESTAMP=$(date +"%Y-%m-%d_%H-%M-%S")
    OutputFile="$DEC_DIR/${VIDEO_NAME}_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.lvc"
   LogFile="$LOGS_DIR/${VIDEO_NAME}_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.log"

    echo -e "\n Rodando encode com SW2 = $current_sw2, QP = $QP" | tee -a "$STATUS_FILE"

    ${ModelEncoder} \
        --width=${WIDTH} --height=${HEIGHT} --format=${FORMAT} --fps=${FPS}
    --qp=${QP} \
        --input_file="${InputFile}" \
        --output_file="${OutputFile}" \
        --base="${BIN_DIR}/${BASE_NAME}" \
        --base_encoder=${BaseEncoder} --encapsulation=nal \
        --cq_step_width_loq_0=${current_sw2} \
        2>&1 | tee -a "${LogFile}"

    DecodedFile="$DEC_DIR/${VIDEO_NAME}_Dec_${BaseEncoderUpper}_LCEVC_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.yuv"
    DecodedLog="$LOGS_DIR/${VIDEO_NAME}_Dec_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.log"

```

```

${ModelDecoder} \
    --base_encoder=${BaseEncoder} --fps=${FPS} \
    --input_file="${OutputFile}" \
    --output_file="${DecodedFile}" \
    --keep_base=true \
    2>&1 | tee -a "${DecodedLog}"

base_file=$(find $SCRIPT_DIR -maxdepth 1 -type f -name "*base.es" | head -n 1)
enhance_file=$(find $SCRIPT_DIR -maxdepth 1 -type f -name "*enhancement.es" | head -n 1)

if [[ -f "$base_file" ]]; then
    base_size=$(stat -c%s "$base_file")
    # mv "$base_file" "$BASES_DIR/${VIDEO_NAME}_Base_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.es"
    rm "$base_file"
else
    base_size=0
    echo "base.es nao encontrado!" | tee -a "$STATUS_FILE"
fi

if [[ -f "$enhance_file" ]]; then
    enhance_size=$(stat -c%s "$enhance_file")
    # mv "$enhance_file" "$BASES_DIR/${VIDEO_NAME}_Enha_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.es"
    rm "$enhance_file"
else
    enhance_size=0
    echo "enhancement.es nao encontrado!" | tee -a "$STATUS_FILE"
fi

output_file_size=$(stat -c%s "$OutputFile")
if (( output_file_size > 0 )); then
    enhancement_ratio=$(awk "BEGIN { printf \"%.4f\", $enhance_size / $output_file_size * 100 }")
    bitrate_kbps=$(awk "BEGIN { printf \"%.2f\", ($output_file_size * 8 * $FPS) / ($NUM_FRAMES * 1000) }")
else
    enhancement_ratio=0

```

```

bitrate_kbps=0
fi

echo -e "\n\t Calculando PSNR..." | tee -a "$STATUS_FILE"
$FFMPEG \
-s:v "${WIDTH}x${HEIGHT}" -pix_fmt "${FORMAT}" -i "${InputFile}" \
-s:v "${WIDTH}x${HEIGHT}" -pix_fmt "${FORMAT}" -i "${DecodedFile}" \
-lavfi psnr="stats_file=${SCRIPT_DIR}/psnr.log" -f null - 2> /dev/null

psnr_avg=$(tail -n 1 ${SCRIPT_DIR}/psnr.log | awk -F'psnr_avg:' '{print $2}' | awk '{print $1}')
rm -f ${SCRIPT_DIR}/psnr.log >> "$STATUS_FILE"

if [[ -z "$psnr_avg" ]]; then
    psnr_avg="N/A"
fi

FileName="${OutputFile##*/}"
echo "${FileName},${current_sw2},${QP},${output_file_size},${base_size},${enhance_size},${enhancement_ratio}%,${psnr_avg},${bitrate_kbps}" | tee -a "$STATUS_FILE" >> "$SizesFile"
echo "${FileName},${current_sw2},${QP},${output_file_size},${base_size},${enhance_size},${enhancement_ratio}%,${psnr_avg},${bitrate_kbps}" >> "$LOCAL_CSV"

echo -e "\n\t SW2=${current_sw2} enhancement_ratio=${enhancement_ratio}% | \
psnr_avg=${psnr_avg} dB | bitrate=${bitrate_kbps} kbps\n" | tee -a "$STATUS_FILE"

rm -f ${SCRIPT_DIR}/*temp* 2>> "$STATUS_FILE"
done

echo -e "\n Script para QP=${QP} finalizado com sucesso!" | tee -a "$STATUS_FILE"

```

I.2 VVC

main-vvc.sh

```

#!/bin/bash

echo -en "\nInicializando Codificacao em VVC... "
for i in {5..1}; do
    sleep 1
    echo -n "${i} "
done
echo -e "\n"

# === Configuracoes compartilhadas ===
# InputFile="/mnt/md0/lucpena/videos/Bosphorus_1920x1080_120fps_420_8bit.yuv"
# InputFile="/mnt/md0/lucpena/videos/SOCCER_352x288_30_orig_02.yuv"
# InputFile="/mnt/md0/lucpena/videos/RaceNight_3840x2160_50fps_8bit.yuv"
# InputFile="/mnt/md0/lucpena/videos/ReadySteadyGo_1920x1080_120fps_420_8bit.yuv"
# InputFile="/mnt/md0/lucpena/videos/Jockey_1920x1080_120fps_420_8bit.yuv"
# InputFile="/mnt/md0/lucpena/videos/city_704x576_yuv420p_60fps_600frames.yuv"
InputFile="/mnt/md0/lucpena/videos/SBTVD/YUV/vc-globo-05_120frames_420p.yuv"

# VIDEO_NAME="Bosphorus"
# VIDEO_NAME="SOCCER"
# VIDEO_NAME="RaceNight"
# VIDEO_NAME="ReadySteadyGo"
# VIDEO_NAME="Jockey"
# VIDEO_NAME="City"
VIDEO_NAME="vc-globo-05_120frames"

# WIDTH=1920
# HEIGHT=1080
# FPS=60
# NUM_FRAMES=600

## Soocer
# WIDTH=352
# HEIGHT=288
# FPS=30
# NUM_FRAMES=300

## City
# WIDTH=704

```

```

# HEIGHT=576
# FPS=60
# NUM_FRAMES=600

## vc-globo-05
WIDTH=3840
HEIGHT=2160
FPS=60
NUM_FRAMES=120

SW2=250
SW2_max=3250
FORMAT="yuv420p"

BaseEncoder="vvc"
Step=500

MAIN_TIME=$(date +"%Y-%m-%d_%H-%M-%S")

QP_list=(22 25 27 30 32 35 37)

ENCODER_SCRIPT="vvc_template.sh"
SCRIPT_DIR=$(dirname "$(readlink -f "$0")")

BaseEncoderUpper=$(echo "$BaseEncoder" | tr '[lower:]' '[upper:]')
SizesFile="$SCRIPT_DIR/${VIDEO_NAME}-${BaseEncoderUpper}-${WIDTH}x${HEIGHT}-${
    MAIN_TIME}.csv"

# Loop para LCEVC
for QP in "${QP_list[@]}"; do
    DIR="QP${QP}"
    mkdir -p "$DIR"
    cp "$ENCODER_SCRIPT" "$DIR/run.sh"

    # Cria script com todas as variaveis embutidas
    cat > "$DIR/config.sh" <<EOF
#!/bin/bash
MAIN_TIME="$MAIN_TIME"
InputFile="$InputFile"
VIDEO_NAME="$VIDEO_NAME"

```

```

WIDTH=$WIDTH
HEIGHT=$HEIGHT
FORMAT="$FORMAT"
FPS=$FPS
NUM_FRAMES=$NUM_FRAMES
SW2=$SW2
SW2_max=$SW2_max
Step=$Step
BaseEncoder="$BaseEncoder"
QP=$QP
BaseEncoderUpper="$BaseEncoderUpper"
SizesFile="$SizesFile"
EOF

# Cria o SizesFile se nao existir
if [ ! -f "$SizesFile" ]; then
    echo "file,sw2,qp,lv_size,base_size,enhance_size,enhancement_ratio,
psnr_avg,bitrate_kbps" > "$SizesFile"
fi

chmod +x "$DIR/run.sh"
chmod +x "$DIR/config.sh"

# Executa em segundo plano
(cd "$DIR" && ./run.sh) &
sleep 2
done

# === Script so com VVC ===
VVC_ONLY_DIR="$SCRIPT_DIR/vvc_only"
VVC_TEMPLATE="$SCRIPT_DIR/vvc_only_template.sh"
mkdir -p "$VVC_ONLY_DIR"

for QP in "${QP_list[@]}"; do
    QPDIR="$VVC_ONLY_DIR/QP${QP}"
    mkdir -p "$QPDIR"

    # Copia template e torna executavel
    cp "$VVC_TEMPLATE" "$QPDIR/run.sh"
    chmod +x "$QPDIR/run.sh"

```

```

# Config file
cat > "$QPDIR/config.sh" <<EOF
#!/bin/bash
MAIN_TIME="$MAIN_TIME"
InputFile="$InputFile"
VIDEO_NAME="$VIDEO_NAME"
WIDTH=$WIDTH
HEIGHT=$HEIGHT
FORMAT="$FORMAT"
FPS=$FPS
NUM_FRAMES=$NUM_FRAMES
BaseEncoder="$BaseEncoder"
QP=$QP
SizesFile="$SizesFile"
EOF
chmod +x "$QPDIR/config.sh"

# Dispara em segundo plano
(cd "$QPDIR" && ./run.sh "$QP") &
sleep 2
done

# === Script com Down/Up sampling ===
DIR="resample"
mkdir -p "$DIR"
cp "resample_template.sh" "$DIR/run.sh"

cat > "$DIR/config.sh" <<EOF
#!/bin/bash
MAIN_TIME="$MAIN_TIME"
InputFile="$InputFile"
VIDEO_NAME="$VIDEO_NAME"
WIDTH=$WIDTH
HEIGHT=$HEIGHT
FORMAT="$FORMAT"
FPS=$FPS
NUM_FRAMES=$NUM_FRAMES
SizesFile="$SizesFile"
EOF

```

```

chmod +x "$DIR/*.sh

(cd "$DIR" && ./run.sh) &

wait

echo -e "\n-----\n Finish \n-----\n"

```

vvc_only_template.sh

```

#!/bin/bash

source ./config.sh

QP="$1"
if [[ -z "$QP" ]]; then
    echo "Erro: QP nao fornecido!"
    exit 1
fi

BaseEncoderUpper=$(echo "$BaseEncoder" | tr '[:lower:]' '[:upper:]')
SCRIPT_PATH="$(readlink -f "$0")"
SCRIPT_DIR="$(dirname "$SCRIPT_PATH")"
STATUS_FILE="$SCRIPT_DIR/status-${QP}.log"
ENC_DIR="/mnt/md0/lucpena/output/main/output"

rm -f "$STATUS_FILE"
echo -e "Iniciando o script VVC Only\n\n" >> "$STATUS_FILE"

SizesFileVVC="$SCRIPT_DIR/${VIDEO_NAME}-${BaseEncoderUpper}-only.csv"

# Checa se o CSV ja existe, se nao, cria com o cabecalho
if [ ! -f "$SizesFileVVC" ]; then
    echo "file,sw2,qp,lvc_size,base_size,enhance_size,enhancement_ratio,
psnr_avg,bitrate_kbps" > "$SizesFileVVC"
fi

```

```

VVC_Encoder="/home/lucpena/apps/LTM/_build_linux/external_codecs/VTM/EncoderApp
"
FFMPEG="/home/lucpena/apps/FFMPEG-VMAF/ffmpeg"

TIMESTAMP=$(date +"%Y-%m-%d_%H-%M-%S")
OutputFile="$ENC_DIR/${VIDEO_NAME}_${BaseEncoderUpper}-Only_QP${QP}.bin"
OutputFileName="${OutputFile##*/}"
OutputYUVFile="${OutputFile%.bin}.yuv"
LogFile="$SCRIPT_DIR/${VIDEO_NAME}_${BaseEncoderUpper}-Only_QP${QP}.log"

CONFIG="/home/lucpena/apps/LTM/_build_linux/external_codecs/VTM/
vtm_encoder_randomaccess.cfg"

echo -e "\n==== Codificando somente VVC QP=${QP} ====\n" | tee -a "$STATUS_FILE"
echo "Codificando com QP=$QP" >> "$STATUS_FILE"

# Codificando para VVC
$VVC_Encoder -c "$CONFIG" --InputFile=${InputFile} --BitstreamFile=${OutputFile} --ReconFile=recon.yuv --SourceWidth=${WIDTH} --SourceHeight=${HEIGHT} --InputBitDepth=8 --OutputBitDepth=8 --InternalBitDepth=8 --FrameRate=${FPS} --QP=${QP} --FramesToBeEncoded=${NUM_FRAMES} --ConformanceWindowMode=1 --InputChromaFormat=420


# Decodificando o VVC para YUV
VVC_Decoder="/home/lucpena/apps/LTM/_build_linux/external_codecs/VTM/DecoderApp
"
FinalFile="${OutputFile%.bin}-recon.yuv"

${VVC_Decoder} -b ${OutputFile} -o ${OutputYUVFile} >> "$STATUS_FILE"

# Calculando PSNR
$FFMPEG -s:v "${WIDTH}x${HEIGHT}" -pix_fmt "$FORMAT" -i "$InputFile" \
-s:v "${WIDTH}x${HEIGHT}" -pix_fmt "$FORMAT" -i "$OutputYUVFile" \
-lavfi psnr="stats_file=psnr.log" -f null - >> "$STATUS_FILE"

psnr_avg=$(tail -n 1 psnr.log | awk -F'psnr_avg:' '{print $2}' | awk '{print $1}')
rm psnr.log >> "$STATUS_FILE"

```

```

output_file_size=$(stat -c%s "$OutputFile")
bitrate_kbps=$(awk "BEGIN { printf %.2f\", ($output_file_size * 8 * $FPS) / (
$NUM_FRAMES * 1000) }")

[[ -z "$psnr_avg" ]] && psnr_avg="N/A"

# Salva no CSV principal e no local
echo "${OutputFileName},265,${QP},${output_file_size},0,0,0%,${psnr_avg},${{
bitrate_kbps}" >> "$SizesFileVVC"
echo "${OutputFileName},265,${QP},${output_file_size},0,0,0%,${psnr_avg},${{
bitrate_kbps}" >> "$SizesFile"

echo " QP=$QP | PSNR=${psnr_avg} dB | Bitrate=${bitrate_kbps} kbps" >> "
$STATUS_FILE"

echo -e "\n\n VVC Only com QP${QP} finalizado!" | tee -a "$STATUS_FILE"

```

vvc_template.sh

```

#!/bin/bash
source ./config.sh

SCRIPT_PATH="$(readlink -f "$0")"
SCRIPT_DIR="$(dirname "$SCRIPT_PATH")"

ModelEncoder="/home/lucpena/apps/LTM/_build_linux/ModelEncoder"
ModelDecoder="/home/lucpena/apps/LTM/_build_linux/ModelDecoder"
FFMPEG="/home/lucpena/apps/FFMPEG-VMAF/ffmpeg"

MAIN_DIR="/mnt/md0/lucpena/output/main"
BASES_DIR="$MAIN_DIR/bases"
DEC_DIR="$MAIN_DIR/dec"
LOGS_DIR="$MAIN_DIR/logs"
BIN_DIR="$MAIN_DIR/bin"
RESULTS_DIR="$MAIN_DIR/results"
STATUS_FILE="$SCRIPT_DIR/status.log"

LOCAL_CSV="$SCRIPT_DIR/${VIDEO_NAME}-${BaseEncoderUpper}-QP${QP}.csv"

```

```

# Checa se o CSV ja existe, se nao, cria com o cabecalho
if [ ! -f "$LOCAL_CSV" ]; then
    echo "file,sw2,qp,lvc_size,base_size,enhance_size,enhancement_ratio,
psnr_avg,bitrate_kbps" > "$LOCAL_CSV"
fi

rm -f "$STATUS_FILE"
echo -e "\n==== Iniciando testes para QP=${QP} ===\n" | tee -a "$STATUS_FILE"

rm -f $SCRIPT_DIR/*.yuv $SCRIPT_DIR/psnr.log 2>> "$STATUS_FILE"

BASE_NAME="${VIDEO_NAME}-QP${QP}-${BaseEncoderUpper}.bin"
if [[ ! -f "${BIN_DIR}/${BASE_NAME}" ]]; then
    echo -e "\n0 arquivo a ${BASE_NAME} nao existe, criando um novo..." | tee -a
"$STATUS_FILE"
${ModelEncoder} \
    --width=${WIDTH} --height=${HEIGHT} --format=${FORMAT} --fps=${FPS}
--qp=${QP} \
    --input_file="${InputFile}" \
    --output_file="$DEC_DIR/base.lvc" --base_encoder=${BaseEncoder}
--encapsulation=nal \
    --cq_step_width_loq_0=${SW2} --keep_base \
    2>&1 | tee -a "$LOGS_DIR/${VIDEO_NAME}_base-gen_QP${QP}_${(date +"%Y-%m-
%d_%H-%M-%S")}.log"

mv "$SCRIPT_DIR/base.bin" "${BIN_DIR}/${BASE_NAME}"

echo -e "Base ${BASE_NAME} criada com sucesso." | tee -a "$STATUS_FILE"
else
    echo -e "Base ${BASE_NAME} ja existente, reutilizando." | tee -a "
$STATUS_FILE"
fi

for (( current_sw2=SW2; current_sw2<=SW2_max; current_sw2+=Step )); do
    TIMESTAMP=$(date +"%Y-%m-%d_%H-%M-%S")
    OutputFile="$DEC_DIR/${VIDEO_NAME}_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.lvc"
   LogFile="$LOGS_DIR/${VIDEO_NAME}_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.log"

```

```

echo -e "\n Rodando encode com SW2 = $current_sw2, QP = $QP" | tee -a "$STATUS_FILE"

${ModelEncoder} \
    --width=${WIDTH} --height=${HEIGHT} --format=${FORMAT} --fps=${FPS}
--qp=${QP} \
    --input_file="${InputFile}" \
    --output_file="${OutputFile}" \
    --base="${BIN_DIR}/${BASE_NAME}" \
    --base_encoder=${BaseEncoder} --encapsulation=nal \
    --cq_step_width_loq_0=${current_sw2} \
2>&1 | tee -a "${LogFile}"

DecodedFile="$DEC_DIR/${VIDEO_NAME}_Dec_${BaseEncoderUpper}_LCEVC_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.yuv"
DecodedLog="$LOGS_DIR/${VIDEO_NAME}_Dec_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.log"

${ModelDecoder} \
    --base_encoder=${BaseEncoder} --fps=${FPS} \
    --input_file="${OutputFile}" \
    --output_file="${DecodedFile}" \
    --keep_base=true \
2>&1 | tee -a "${DecodedLog}"

base_file=$(find $SCRIPT_DIR -maxdepth 1 -type f -name "*base.es" | head -n 1)
enhance_file=$(find $SCRIPT_DIR -maxdepth 1 -type f -name "*enhancement.es" | head -n 1)

if [[ -f "$base_file" ]]; then
    base_size=$(stat -c%s "$base_file")
    # mv "$base_file" "$BASES_DIR/${VIDEO_NAME}_Base_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.es"
    rm "$base_file"
else
    base_size=0
    echo "base.es nao encontrado!" | tee -a "$STATUS_FILE"
fi

```

```

if [[ -f "$enhance_file" ]]; then
    enhance_size=$(stat -c%s "$enhance_file")
    # mv "$enhance_file" "$BASES_DIR/${VIDEO_NAME}_Enha_LCEVC_${BaseEncoderUpper}_SW2-${current_sw2}_QP${QP}_${TIMESTAMP}.es"
    rm "$enhance_file"
else
    enhance_size=0
    echo "enhancement.es nao encontrado!" | tee -a "$STATUS_FILE"
fi

output_file_size=$(stat -c%s "$OutputFile")
if (( output_file_size > 0 )); then
    enhancement_ratio=$(awk "BEGIN { printf \"%.4f\", $enhance_size / $output_file_size * 100 }")
    bitrate_kbps=$(awk "BEGIN { printf \"%.2f\", ($output_file_size * 8 * $FPS) / ($NUM_FRAMES * 1000) }")
else
    enhancement_ratio=0
    bitrate_kbps=0
fi

echo -e "\n\tCalculando PSNR..." | tee -a "$STATUS_FILE"
$FFMPEG \
    -s:v "${WIDTH}x${HEIGHT}" -pix_fmt "${FORMAT}" -i "${InputFile}" \
    -s:v "${WIDTH}x${HEIGHT}" -pix_fmt "${FORMAT}" -i "${DecodedFile}" \
    -lavfi psnr="stats_file=${SCRIPT_DIR}/psnr.log" -f null - 2> /dev/null

psnr_avg=$(tail -n 1 ${SCRIPT_DIR}/psnr.log | awk -F'psnr_avg:' '{print $2}' | awk '{print $1}')
rm -f ${SCRIPT_DIR}/psnr.log >> "$STATUS_FILE"

if [[ -z "$psnr_avg" ]]; then
    psnr_avg="N/A"
fi

FileName="${OutputFile##*/}"
echo "${FileName},${current_sw2},${QP},${output_file_size},${base_size},${enhance_size},${enhancement_ratio}%,${psnr_avg},${bitrate_kbps}" | tee -a "$STATUS_FILE" >> "$SizesFile"

```

```

echo "${FileName},${current_sw2},${QP},${output_file_size},${base_size},${enhance_size},${enhancement_ratio}%,${psnr_avg},${bitrate_kbps}" >> "$LOCAL_CSV"

echo -e "\n\t SW2=${current_sw2} enhancement_ratio=${enhancement_ratio}% | psnr_avg=${psnr_avg} dB | bitrate=${bitrate_kbps} kbps\n" | tee -a "$STATUS_FILE"

rm -f $SCRIPT_DIR/*temp* 2>> "$STATUS_FILE"
done

echo -e "\n Script para QP=${QP} finalizado com sucesso!" | tee -a "$STATUS_FILE"

```

I.3 Resample

O script para *Resample* é o mesmo para os dois *mains*.

resample_template.sh

```

#!/bin/bash

source ./config.sh

SCRIPT_PATH=$(readlink -f "$0")
SCRIPT_DIR=$(dirname "$SCRIPT_PATH")
STATUS_FILE="$SCRIPT_DIR/status.log"

ModelEncoder="/home/lucpena/apps/LTM/_build_linux/ModelEncoder"

rm $STATUS_FILE
echo -e "Iniciando Down+Up Sampling...\n" > "$STATUS_FILE"

SizesFileResample="$SCRIPT_DIR/${VIDEO_NAME}-Resample-Only.csv"

# Checa se o CSV ja existe, se nao, cria com o cabecalho
if [ ! -f "$SizesFileResample" ]; then
    echo "file,sw2,qp,lvc_size,base_size,enhance_size,enhancement_ratio,psnr_avg,bitrate_kbps" > "$SizesFileResample"

```

```

fi

FFMPEG="/home/lucpena/apps/FFMPEG-VMAF/ffmpeg"

Downsampled="$SCRIPT_DIR/${VIDEO_NAME}_downsampled.yuv"
Upsampled="$SCRIPT_DIR/${VIDEO_NAME}_upsampled.yuv"

echo -e "\nDownsampling...\n" | tee -a "$STATUS_FILE"
${ModelEncoder} --downsample_only=true --format=${FORMAT} -w ${WIDTH} -h ${HEIGHT} \
-i ${InputFile} -o ${Downsampled} >> "$STATUS_FILE"

echo -e "\nUpsampling...\n" | tee -a "$STATUS_FILE"
${ModelEncoder} --upsample_only=true --format=${FORMAT} -w ${WIDTH} -h ${HEIGHT} \
-i ${Downsampled} -o ${Upsampled} >> "$STATUS_FILE"

echo -e "\nCalculando PSNR...\n" | tee -a "$STATUS_FILE"
$FFMPEG -s:v "${WIDTH}x${HEIGHT}" -pix_fmt "$FORMAT" -i "$InputFile" \
-s:v "${WIDTH}x${HEIGHT}" -pix_fmt "$FORMAT" -i "$Upsampled" \
-lavfi psnr="stats_file=psnr.log" -f null - >> "$STATUS_FILE"

psnr_avg=$(tail -n 1 psnr.log | awk -F'psnr_avg:' '{print $2}' | awk '{print $1}')
rm psnr.log >> "$STATUS_FILE"

output_file_size=$(stat -c%s "$Upsampled")
bitrate_kbps=$(awk "BEGIN { printf "%.2f\", ($output_file_size * 8 * $FPS) / (\$NUM_FRAMES * 1000) }")

# Salva no CSV principal e no local
echo "${VIDEO_NAME}-Resample,0,0,$output_file_size,0,0,0%,${psnr_avg},0" >> \
$SizesFile
echo "${VIDEO_NAME}-Resample,0,0,$output_file_size,0,0,0%,${psnr_avg},0" >> \
$SizesFileResample

[[ -z "$psnr_avg" ]] && psnr_avg="N/A"
echo " Down-Up Sampling  PSNR=${psnr_avg} dB" >> "$STATUS_FILE"

echo -e "\n\n Downsampling + Upsampling finalizado!" | tee -a "$STATUS_FILE"

```

```
rm $SCRIPT_DIR/*.yuv
```